LLM Problem Analysis and DSM Deep Semantic Model

CHEN Feng **Nancy Chen**

Abstract

This paper analyzes the main problems of the current LLM and proposes specific solutions, pointing out the fact that: the expression and computation of the conceptualized structural model combined with probability is the key, and provides a brief explanation of the related technology-Deep Semantic Model (DSM), and finally enumerates the direction of the subsequent key work.

Key words: LLM; GPT; Hallucination; DCN; Dynamic Cognitive Network; DSM; Deep Semantic Model; Interpretable; Conceptualization; Structurization; AGI;

content

1. Introduction

Nowadays, the development of LLM has greatly improved the technical level of natural language processing, revealing that AI technology has great capabilities and application prospects, and will bring a number of positive impacts to human society, which has already formed a consensus in the industry.

However, in the meanwhile, many critical problems of LLM technology have also been exposed in the ever-deepening research and application, which have caused obvious obstacles to the further enhancement of the technology and the full realization of the application value. There are also considerable views in the industry that the current LLM is not the ultimate solution to realize AGI.

Taking a typical representative product ChatGPT as an example, this paper provides an in-depth analysis of the main problems of similar LLMs and puts forward fundamental solutions or directions.

This paper also provides a brief introduction to the DSM deep semantic technology, elaborating on the key points of its basic theory, model architecture, realization method, and current results. It analyzes the model of the technology's solution to the above problems, as well as the model of collaboration between the technology and LLM to achieve better technical solutions and products, and points out the direction of the subsequent key work.

It is important to note that the connotations of conceptual terms are difficult to define precisely

and are constantly changing. In this paper, LLM refers to the commonly accepted definition in the current industry: a model that employs a deep neural network architecture, which is trained by automated machine learning on a large amount of corpus and forms a black-box structure containing a large number of non-conceptualized connections and parameters, computing on natural language in an end-to-end manner. This paper analyzes GPT as an example and points out that most of the issues are applicable to other current LLMs, and that a small number of issues may not be applicable to some other LLMs, but do not affect the overall conclusions.

2. Analysis and Resolution of GPT-related Issues

2.1 Interpretability issue: representation and computation of conceptualization structures

Interpretability can be defined as the ability to explain or present the behavior of a model in understandable terms for human. Interpretability should not be just a measure of a system, but a goal of system implementation that is as important as functional effectiveness. As people study all kinds of sciences to construct interpretable systems, AGI serves to replicate and enhance the ability of human thinking, and interpretability is likewise the core goal of AGI. Even by the standard of result-only theory, the ability to interpret a system determines the ability to decompose, adjust, and control the system, as well as the upper limit of the ultimate functional effect of the system.

At present, LLM suffers from the problem of poor interpretability, which bottlenecks the further improvement of its capabilities. In addition, the problems of relying on massive data, massive repetitive training, and catastrophic obliviousness are essentially a manifestation of this root problem.

The most effective way to solve interpretability problems is conceptualization and structuralization.

Conceptualization is to define human-understood concepts as basic elements that make up the system. Take GPT3 as an example, 12,288 dimensional vectors are used to express the basic information, which are mainly learned automatically by machines and are not aligned with human-understandable concepts. Conceptualization is to define human-understood concepts as basic elements that make up the system. Take GPT3 as an example, 12,288 dimensional vectors are used to express the basic information, which are mainly learned automatically by machines and are not aligned with human-understandable concepts[1]. Assuming that these 12,288 vector dimensions can be equivalently converted to another 12,288[2] human-understandable concepts, the goal of conceptualization is achieved to some extent. It is of course desirable to achieve this goal if it can be accomplished by automatic bottom-up machine learning alone, but if not, then a combination of top-down human design is of great necessity.

[1] Each vector dimension is actually a mixture of multiple concepts. Regardless of the method used, if interpretability is achieved, it also means that these vectors are disassembled and reorganized and aligned with human-understood concepts. At this point it becomes inevitable to perform transformation optimization.

[2] Theoretically, it is certainly possible to accomplish an equivalent transformation with another 12,288 dimensions. But the probability is that it can be optimized to fewer dimensions.

Conceptualization is accompanied by structuralization. In the example of GPT, the connections that compute vectors are also non-conceptualized and have only probabilistic computational parameters without semantic information. Conceptualization and semanticization of many connections is also important to form a conceptual structure that combines description and computation.

At the same time, Transformer is fully connected, which is very suitable for the initial first exhaustive discovery of all possible knowledge. However, the fixed structure also means that after effective knowledge is learned, a large amount of invalid knowledge with probability parameter close to 0 still takes up space and computational resources. Conceptualization and structuralization also involves pruning, merging and optimizing concepts and structures.

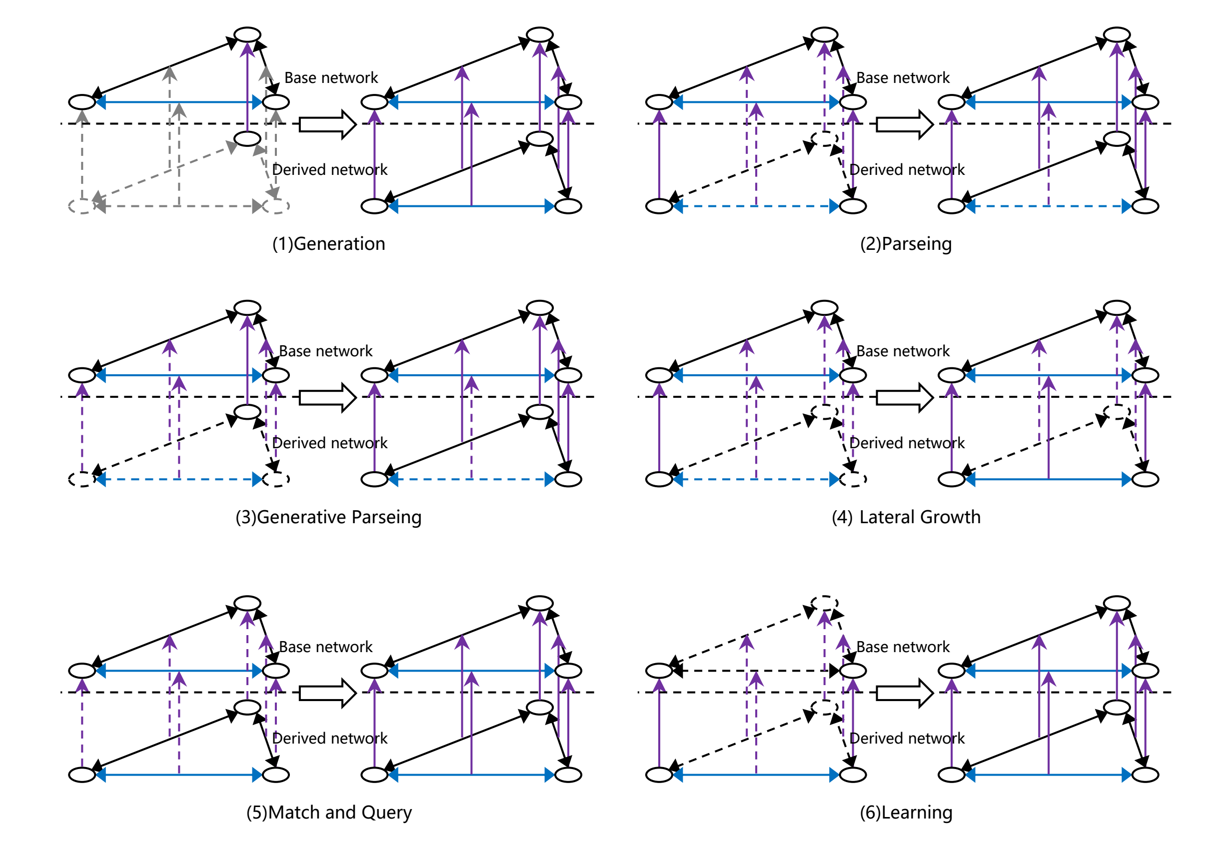
The statement "Intelligence is compression" is correct. The nature of the human mind is to process a nearly infinite amount of information with a finite brain capacity. The essence of human thinking is to process nearly infinite information with a finite brain capacity, and the key still lies in specific methods. Whereas combination and generalization are efficient methods for compressing information. More adequate conceptualization and structuralization will increase the compression rate of effective knowledge to a higher level, and explanatory and its related problems will no longer be a problem.

At this point, it is very important to work on constructing a system of wide-area base concepts (including conceptualization vectors) and structural systems. This can be used to explain LLM and compensate for its lack of semantic structure, and serve as a basis for the development of more powerful AI systems.

2.2. Incomplete algorithmic system issue: designing complete algorithmic systems

The incomplete algorithmic system is also one of the more prominent fundamental theoretical flaws of GPT, to which many difficult problems and results can be attributed.

With the viewpoint of DCN Dynamic Cognitive Networks[1], the algorithms of understanding, querying, reasoning, generating, and learning (Fig. 1) are important basic algorithms for realizing AGI, which are not interchangeable with each other, and can be combined to solve various computational tasks in a complete way.



Solid line: elements that have grown; **dashed lines:** elements that objectively exist and have not grown; **gray dashed lines:** elements that objectively do not exist

Black: concept and longitudinal relation; **purple:** belong to relation; **blue**: lateral relation;

Fig.1：Typical Algorithm Schematic Diagram

GPT mainly uses the generalized generation algorithm[1], which refers to a known number of characters to calculate the next character (strictly speaking, Token). The basic principle can be summarized as:

1. The vector of the target character is computed separately from the vector of each character in the known character sequence.

2. Each known character is superimposed on the target vector calculated for the target character.

3. The superimposed vectors of the target character are compared with a character dictionary and the best match is taken as the result.

This algorithm works well for generative tasks, but lacks key capabilities such as complete structured hierarchy computation and probabilistic collapse computation, and is not an equivalent

[1] Note: Generation in DCN refers specifically to the top-down computation of the derived network from the base network in set dimension, while generation in GPT refers generally to the computation of the next character from the previously known string (which actually corresponds to "growth" in DCN). To avoid confusion, the former is referred to as "narrow generation" and the latter is referred to as "generalized generation" when necessary.

replacement for other algorithms. Forced substitution cannot make the effect of various tasks converge to the ideal state. The fundamental solution is to design a more complete algorithmic system, which should be designed around the above conceptualized structure.

A selection of issues are analyzed more specifically below.

2.3. Issues such as hallucination: choosing the right algorithm

Generally speaking, LLM computations that produce results that do not meet people's expectations and standards are collectively referred to as hallucinations, but different hallucinations have different causes. Here we mainly discuss the most essential type of hallucination, i.e., generative hallucination, which is summarized by the technical principle of "probability loss caused by the generation of the base network to the derived network" and the problem of " probability as necessity". The details are analyzed as follows:

- The difference between comprehension, generation, and equivalence calculation

First, the rules for comprehension, generation, and equivalence of these computations are different. In the system of DCN, these algorithms rely on [belong to relation] and [equivalence relation] for implementation. We first assume that some knowledge [Lin Daiyu belongs to a person], [Jia Zheng belongs to a person], [Granny Liu belongs to a person], and [People insulting people] already exists in the system[1] (note: where the probability of [People insulting people] being factual is < 1, because an insulting event between any two people is probable, but not factual). Hence the algorithms are as follows:

Comprehension: understanding from [Lin Daiyu insulting Jia Zheng] to [People insulting people], there is no loss of probability in the bottom-up calculation.

Equivalence: copying equivalently from [Lin Daiyu insulting Jia Zheng] to [Lin Daiyu insulting Jia Zheng], which is completely equivalent.

Generation: deriving [Lin Daiyu insulting Jia Zheng][2] from [People insulting people]. The former is the base model and the latter is the derived model, and the concepts corresponding to the two patterns have the [belong to] relation respectively. The derived model will first directly inherit the parameters of the base model including the probabilities, i.e., the factual probabilities < 1. Narrow generation is a top-down computation on the ensemble dimension, and direct generation out of nowhere will also result in generating probability losses. Unless more information is incorporated to influence and adjust the new probabilities.

Learning: When acquiring factual knowledge such as "Lin Daiyu insulting Granny Liu" for learning, the first step is to understand it, construct the new knowledge [Lin Daiyu insulting Granny Liu] and establish a derivative relationship with [People insulting people], and set the factual probability reloading = 1, so that the new knowledge is deposited into the system to complete the learning

---

[1] Note: In the novel [Dream of the Red Mansion], it happened that the main character Lin Daiyu insulted Granny Liu, but there was no such thing as Lin Daiyu insulting Jia Zheng.

[2] This paper has repeatedly referred to the concepts of knowledge hierarchy, knowledge and data. Here is a typical example: "people insulting people" is a higher-level knowledge, and "Lin Daiyu insulting Jia Zheng" is a lower-level knowledge, i.e., closer to data. In this way, the knowledge level or conceptual level has a clear criterion and can be parameterized.

and memorization. Obviously, the knowledge learned from the reliable information source is different from the information generated by the system itself, especially in the factorial probability.

Overall, the relations of belong to, derivation, and equivalence have very different semantics. If expressed in terms of probability, it is also necessary to distinguish between the different directions of the projected probability of the existence of <1 and =1 of the essence of the difference. There is only one one-way probability calculation in GPT, and the result is based only on the relative size of the probability, and the underlying algorithm is fundamentally difficult to realize the task of calculating probability=1.

- Difference between generation task and query task

Whereas humans propose tasks for different purposes. Understanding, querying, and generating are completely different types of tasks, with different criteria for determining the correctness of the results, which are completely clear in themselves.

The [generation] task follows the criterion of probabilistic possibility and does not require probability=1 factuality. The [generation] algorithm should be used, e.g., [make up a story about " insulting "], then the knowledge of [people insulting people] can be used to derive an infinite number of outcomes such as [Zhang San insulting Li Si] [Lin Daiyu insulting Zhang San] [Lin Daiyu insulting Jia Zheng]...etc., all of which satisfy the requirements of the task.

The generalized generation algorithm of GPT is well suited for this narrow generation task[1]. Since generalized knowledge like [people insulting people] is obtained through training and learning, the use of vector computation to generate derivatives of this knowledge can produce a variety of results similar to those described above to satisfy the requirements of the task.

The [query] task, on the other hand, follows probabilistic determinism, i.e., probability=1 factuality. Correct processing requires the use of the [query] algorithm, which is the pattern matching algorithm.

For example, for the task of [querying an " insulting" story], the correct approach should be to use the question as a template for pattern matching on factual knowledge. If the match is successful, the factual knowledge obtained, e.g., [Lin Daiyu insulting Granny Liu], will be copied and output equivalently. If the result is not satisfied, the answer "the query cannot match the result" should be given. The results calculated in this way are theoretically completely stable and reliable, and problems can be accurately traced and corrected.

(In the example here, the knowledge of [people insulting people] in the system will also get matched, but this one is high in the knowledge hierarchy and has to be excluded from the results. This processing is also based on a strict theoretical foundation.)

And for GPT, it is still executed with the same generation algorithm. If the sentence [Lin Daiyu insulting Jia Zheng] is generated based on the same aforementioned sentence [people insulting people], the result obtained with the generation task does not meet the requirements of the query task.

---

[1] The existence of multiple correct outcomes (people: Jia Zheng, Lin Daiyu, Jia Baoyu...) for top-down narrow generation on the set dimension does not require a uniquely deterministic outcome and is more fault-tolerant. This is an important reason why various generative AIs are currently enjoying greater success.

This is the root cause of this typical hallucination.

With the training, GPT affects the probability of selecting different knowledge as the [base pattern] for generation based on the task vocabulary such as [query] [make up], so that the knowledge [Lin Daiyu insulting Granny Liu] has a higher probability of selection than the knowledge [people insulting people] as the [base pattern] for [generation]. From a certain point of view, this way is differentiating the tasks, but this probability adjustment lacks the support of rigorous computational rules, and the probability calculations and choices for different knowledge lack reliability (even treating the user's question itself as a [base pattern] and deriving it directly as an answer, which often happens in practice) still results in various phantom errors), remaining a source of all kinds of hallucination errors.

In addition, the query task requires that every parameter contained in the multi-parameter question pattern is matched by the knowledge pattern, and involves performing the [And] computation to ensure the wholeness of the pattern, which is difficult to satisfy with the only basic algorithm of the GPT, the generation algorithm.

In real-world scenarios, a task will have multiple subtasks nested within it. These subtasks are of different types and will not all have explicit task indicator keywords. For example, a generation task may have multiple parameters nested within it that require factual queries. Without in-depth semantic analysis and then choosing the right algorithm to process them separately, it will be difficult to avoid the generation of hallucinations. Moreover, the hallucination information is likely to be so hidden that the user will not be able to detect it, which may lead to more serious consequences.

As a result, the solution is to work both ways:

- Accurately differentiate the types of different tasks and subtasks through deep semantic analysis.
- Select the correct basic algorithm for different task types.

2.4 Issues such as computation of expressions: structured hierarchical expressions and computations

Hierarchical representation and computation of knowledge is very important. The hierarchical structure of images is remarkable, and the semantics of language is entirely structured in layers - although such layers do not always correspond exactly to the organization of language. GhatGPT, on the other hand, lacks the ability to represent and compute rigorously structured hierarchies.

At present, the algorithm of ChatGPT itself cannot solve a simple expression computation. It may be able to correctly answer the expression 35+62=97, but the essence of the implementation is to completely match the sentence 35+62=97 in a huge corpus, not a real mathematical calculation. So if you ask " 23456789+9876543=how much", there is a high probability that you will not get the correct result[1].

[1] The fact that GPT can invoke plug-ins to solve mathematical computation problems does not address the shortcomings that exist on their own. If basic capabilities such as expression computation cannot be built into the model, the level of integrated intelligence to perform composite tasks will be extremely limited.

The fundamental problem here is that the GPT lacks a true hierarchical representation of information[1]. "23456789+9876543" is obviously composed of two levels and two numbers as a whole at the lower level, to participate in the calculation of addition at the higher level. Instead, GPT always disassembles these characters and looks at a flat one-dimensional sequence of characters, each of which is involved in the probability calculation of the next character. In this case the probability calculation is ~~simply~~ impossible to get the correct result.

As we can see from this example, GPT is also very unreasonable for regular natural language computation model. First of all, it is a great waste of computational resources, which is not only energy-consuming, but also has the limitation of context length. Under the hierarchical computation model, each character in a 100,000-word article should not be computed for all other characters, but should be limited to a local range. The amount of computation that grows with the context should be close to linear, not geometric, like that of the human brain.

Therefore, it is of great importance to achieve a truly hierarchical expression and computation of knowledge. This is not only to solve such local problems as expression computation, or optimization of computational performance, but definitely one of the important basic indexes for the realization of AGI!

The above example also exemplifies another problem: vectors are not suitable for the exact expression of numbers nor are they difficult to express deterministic concepts. From the point of view of hierarchical structure, the root of a hierarchical structure can be represented by an identified concept to precisely represent the whole structure, an ability that vectors do not have. Therefore, one cannot represent everything with vectors alone, and an identified representation of the concept is also necessary. The integration of the two is a valuable topic.

2.5 Knowledge inconsistency issue: bidirectional relationship representation and calculation

GPT suffers from serious knowledge inconsistencies.

As an example, "One study found that there is a [reversal curse] on large models that prevents them from reasoning that [B is A] even after learning that [A is B]. For instance, when we teach a model that [Washington was the first president of the United States], it does not automatically answer [Who was the first president of the United States?] , unless additionally taught that [The first president of the United States was Washington]."

This problem arises simply because of the fact that the GPT is a neural network for one-way probabilistic expression and computation, and it employs the projection of characters from the front to the back. [Washington was the first president of the United States] and [The first president of the United States was Washington] would be treated as two completely different connections and computations, rather than one piece of holistic knowledge.

This problem exposes the essential fact that no real structured knowledge is learned in such a way as in GPT! Adding more morphologically different but inherently redundant information can

[1] The multi-layered neural network structure of GPT implicitly expresses certain hierarchical information. However, the capability cannot be compared to a truly semanticized hierarchical knowledge structure, and it is difficult to achieve more powerful nested computation of hierarchical combinations.

mitigate it in terms of effect, but it will not help to improve the overall level of intelligence.

Obviously, this consistency of knowledge is not a problem for human beings. Knowledge of a whole should be expressed and memorized as a whole and can be flexibly applied in different forms.

Therefore, the correct solution is to model the human approach. Specifically, the first step is to replace the unidirectional function computation with a semanticized relation with bidirectional probabilities. This kind of relation can express complete semantic knowledge and can also be used to perform probabilistic computation, which is a more reasonable, effective and complete knowledge expression. Furthermore, the semantic structure of such relations can realize the "scene-fitting" expression and computation mode expressed by DCN theory.

This model may well be the key to solving AGI.

2.6 Weaknesses in reasoning issue: structured reasoning expression and computation

According to DCN, roughly[1] speaking: comprehension and generation mainly refer to the vertical computation of a tree network from bottom to top and from top to bottom, the depth of computation is shallow and relatively fixed, which is the foundation of intelligence. Reasoning, on the other hand, mainly refers to the horizontal transformation between multiple tree networks, and the depth of computation can be very deep and range expansive, which determines the upper limit of intelligence[2].

Compared to the past situation where there was a complete lack of reasoning ability shown in natural language form, GPT has realized a considerable degree of reasoning ability, which is a great progress. However, the current level of reasoning is still relatively weak by higher intelligence standards. To continuously improve the reasoning ability and intelligence level, it is not only a matter of increasing the corpus and training workload, but also the fundamental problems of technical principles and architecture need to be solved.

- GPT's representation of inference knowledge, like other knowledge, is a black-box structure that lacks stability and reliability and is difficult to adjust with precision. The problem becomes more pronounced as more inference knowledge is added. Consistent with the previous review, a more conceptual and structured way to express reasoning knowledge and perform reasoning computation is a truly effective solution. Assuming that learning training in the form of chains of thought, etc., has allowed the system to learn reasoning knowledge to a certain degree of effectiveness, optimizing this knowledge into more essentially structured representations as well as expanding the parameters that matter (probabilities, etc.) will inevitably lead to even better results. Of course, the reasoning structure is more complex than the simple structure, and is a difficult problem that has not been effectively solved by techniques such as traditional knowledge graphs, thus requiring a better theory of structural design to solve it.

---

[1] Various computations exist in close relationship, e.g., comprehension and generation are also accompanied by reasoning, which can only be roughly delineated here.

[2] Derivative structures (set dimension) or nested structures (domain dimension) with less than 10 layers are expressive enough to be processed quickly, which is close to "fast thinking"; Advanced reasoning will have very deep chains, with uncertainty in each link, which will be reflected as "slow thinking".

- Non-hierarchical and holistic issues: As previously discussed, reasoning computations require stricter requirements of hierarchical and holistic nature. A single atomic reasoning computation should be a complete conversion from one pattern to another, and should not be reasoned character by character. The latter is not only computationally intensive, but also prone to the problem of inconsistent information.
- Bidirectional reasoning flaw: As previously discussed, the reasoning of the GPT is unidirectional and must be elevated to the overall structure of bidirectional reasoning to express and compute.
- Multi-branch, multi-level complex reasoning: reasoning in real-world scenarios involves multi-branch, multi-level reasoning in the face of a wide domain of information.
- The maximum probability branch selected by the first level of reasoning may not produce the optimal result after multiple levels of reasoning. Without formalized expressions, decomposable combinations, dynamic parametric reasoning structures and more flexible multi-way reasoning and retrospectively adjustable models, it is difficult to obtain the desired results.

2.7 Knowledge and data mixing issue: hierarchical representation of knowledge and data

GPT does not have a hierarchical definition of knowledge and cannot effectively distinguish between knowledge and data, and manages knowledge and data mixed together. And currently when using a model such as PROMPT and combining it with external information, both the internal and external information of the model are actually mixed with knowledge and data respectively. It does not really separate knowledge and data, but rather complicates the problem.

Due to the difficulty of isolating the correct knowledge for effective sharing, different forms of iterative model training are being performed in the industry, generating many black-box knowledge bases with large amounts of redundancy but no uniformity, which continues to produce a waste of duplicated resources.

The solution is to express and manage knowledge and data hierarchically. The computation can be seamlessly integrated, but the management structure can be flexibly decomposed. The importance and authority of each piece of knowledge can also be precisely defined and managed. Higher-level deterministic knowledge can be fully shared, while lower-level data can be stored flexibly, and different versions of knowledge and data that cannot form a consensus can also be maintained. In this way, a more reasonable knowledge and data maintenance system can be constructed.

Hierarchical management of knowledge and data also facilitates assistance in resolving data copyright issues at the technical level. True higher-level knowledge is maintained at a small scale, and there are no copyright issues with this human consensus knowledge. A large amount of lower-level knowledge (e.g., all kinds of news) should be embodied as data stored independently, and can be refined to identify each piece of knowledge in terms of copyright, and can even generate new business models adapted to the age of intelligence.

2.8 Knowledge learning issue: multi-level incremental learning models

LLM is an overall black-box structure that cannot be broken down into individual pieces of knowledge for adjustment, nor can it effectively distinguish between correct and incorrect knowledge. This leads to problems such as catastrophic oblivion when learning new knowledge that may undermine existing knowledge.

The solution is to realize real-time and incremental learning based on knowledge hierarchy. Since each piece of knowledge can be split, it can form an efficient pattern similar to human learning knowledge. The already explicit knowledge is fixed first, and then only the new knowledge is learned incrementally, which is more stable and reliable to continuously accumulate knowledge.

The learning of truly layered knowledge will be more efficient. There is more knowledge at the shallow level, and the learning actions are of more high frequency, but only the more peripheral knowledge base needs to be modified. If higher-level knowledge needs to be adjusted, then the higher-level base knowledge base is passed on for modification. Typically, the higher levels have less knowledge and more low-frequency learning adjustments.

2.9 Alignment of uncertain information issue: conceptual representation of certainty

Natural language has complex ambiguity, and the same natural language word often corresponds to multiple different semantics, which has been the core problem of natural language processing[1]. The key to solving the ambiguity problem is to fully utilize the contextual information, specifically the individual words to project each other, which is essentially what GPT does: GPT internally disassembles the vocabulary into Token and then converts it into vectors, and lets the vectors extrapolate each other to eliminate ambiguities. Eventually, based on the result vectors are converted to Token and vocabulary again, indirectly inferring from the input and output representations. The accuracy of natural language understanding is excellent[2].

However, the hidden semantic concepts formed by GPT comprehension, especially the overall structure, are difficult to express. There is not even a complete set of semantic standards that can be targeted, so the GPT output is still natural language. This causes the problem that semantic concepts as well as semantic structures are difficult to be aligned, and it is difficult to realize reliable information docking and sharing among multiple systems and models by relying only on the form of natural language.

It is well known that the exchange of information between systems starts with ensuring the consistency of information standards. Therefore, a global semantic concept and structure is necessary even from an engineering point of view only. Only with deterministic semantic expressions can information be effectively shared and transferred between systems, and any two systems can be

---

[1] If all natural languages can always be expressed as standard structures that are rich enough in information and free of ambiguity, then many complex intelligent tasks can be realized on this basis, even with the use of stacked manual development.

[2] It is still questionable whether the comprehension of the first session reaches the standard of human comprehension when judged from the results of a multi-session end-to-end dialog. Traditional machine translation techniques clearly do not really understand the semantics, but can output results that are not too far off all the same. In a sense, machine translation has a relatively unique standard of results, and errors are easily detected by humans, whereas problems with diffuse generated dialogs are harder to detect.

directly interfaced without the need for GPT. Moreover, any two systems can be directly interfaced without the requirement of GPT, and problems such as task decomposition and combination, multi-technology integration, and long-term memory access can be better solved.

2.10. Bias and jailbreaking issue: semantics-based control of absolute information

Information control issues such as bias, jailbreaking, and security are also challenges faced by LLM technology.

Nowadays, it is common to try to solve the problem by constant training fine-tuning, and prompt. However, this model hardly guarantees stable and reliable results and eventual convergence. Information interacts with each other, e.g., assuming that a prompt works, another prompt that is peripherally spiked again can also request the system to turn off the effect of the previous prompt[1]. A probabilistically optimal model cannot solve the problem of probabilistic absolutes.

The solution is in two ways:

- The first step is still to define semantic concepts and structures. The semantics of [bias], [sensitive information], etc. are also accurately defined, and accurate semantic parsing and categorization of both input and output information can be achieved.
- Then the next step is to have a precise semantic basis. Reliable information control can be inserted at any point in the process, giving absolute control over specific processing rules (probability set to =1, no other computation allowed to override).

3. Introduction to the DSM Deep Semantic Model

DSM (Deep Semantic Model) is a specific implementation of DCN (Dynamic Cognitive Network) theory for language semantic processing.

Compared with traditional knowledge graphs, DSM can realize important capabilities such as deep semantic expression, complete semantic expression, hierarchical semantic expression, algorithmic closed-loop system, probabilistic expression and computation, and docking of natural language, forming a complete linguistic semantic expression and computation system. It has the potential to become an independent and complete intelligent system. Comparatively speaking, traditional knowledge graphs are usually more suitable for constructing thematic databases to provide data for intelligent systems but difficult to be used as an autonomy-driven intelligent system.

DSM has extensive content, and this paper will only provide a brief introduction to the technical points.

3.1. Deep semantic structure

The basis of the deep semantic model lies in the definition of its unique DSM structure, which

[1] In fact, can GPT make any strict distinction at all between the rules that the system designers have trained for it and the commands that ordinary users give to it?

adopts the two-dimensional multi-level tree network structure proposed in the DCN theory and is optimized according to the characteristics of the language.

Deep semantic structure is also a structure that integrates expression and computation. The structure itself expresses the conceptualized semantic knowledge and also describes the basic rules and parameters of computation. Various computations are embodied in various creations, combinations and transformations of the structure relying on its own semantics and parameters.

3.2 Separation of semantics and language

DSM completely separates the semantic model from the language model. Semantics is independent of natural language, and the two are interchanged through comprehension and generation algorithms. "Use language for extrinsic expression and semantics for intrinsic expression and computational thinking."

Semantic and linguistic transformations are described and computed mainly through two [owning] relations:

- Semantic concepts has linguistic morphology.
- Semantic roles has linguistic roles.
- Linguistic is understood to be the same set of relational structures shared by the reciprocal operations of semantics and semantically generated linguistic.

3.3 Hierarchical Semantic Knowledge and Data System

Following a basic principle - "human knowledge systems are hierarchical" - DSM builds a multi-level semantic knowledge system.

The higher-level knowledge is more basic and important, and is the basis for understanding and expressing lower-level knowledge, while the amount of higher-level knowledge is more limited. The interpretability and computability of a semantic model is mainly represented by the topmost level of knowledge. Such topmost knowledge includes [concepts] [entities] [relationships] [roles] [existence] [measures] [degrees] [sets] [intervals] [comparisons] [sequences] [space] [time] [things] [events] [event roles] [expressions] [equations]... etc. The interpretability and computability of a semantic model is mainly represented by the topmost level of knowledge. Such topmost knowledge includes [concepts] [entities] [relationships] [roles] [existence] [measures] [degrees] [sets] [intervals] [comparisons] [sequences] [space] [time] [things] [events] [event roles] [expressions] [equations]... etc.

The vast amount of knowledge in the lower and middle levels is much more extensive and theoretically infinitely expandable. But it is all interpreted and computed using the higher level knowledge. In principle, it is no longer necessary to implement different algorithms for different knowledge[1].

The idea of a hierarchy of knowledge also applies to the division of knowledge and data. Data is seen as a lower-level of knowledge, and the two are theoretically completely isomorphic and can

[1] Of course, specific business applications can also be realized by expanding algorithms for specific knowledge as needed.

be seamlessly integrated.

In terms of storage management, "separation of knowledge and data" can be realized, since lower-level knowledge is unidirectionally dependent on higher-level knowledge, different levels of knowledge and data can be stored separately. For computational processing, higher-level knowledge must be loaded, while mid- and lower-level knowledge and data can be loaded dynamically on demand, and can be designed in a variety of specialized structures (e.g., relational databases) for optimal expression as well as in natural language, which can be viewed as a compressed form of deep semantics.

3.4. Inheritance, overloading and aggregation

DSM uses belong to relation and inheritance mechanism to realize the hierarchical representation of knowledge. The lower-level knowledge first inherits the derived network of the higher-level knowledge by default, thus inheriting all the information of the base network. Instead, in response to changes in the information of the derived network with respect to the base network, an overloading definition is performed to modify the changed information (including probability distribution parameters, etc.).[1]

DSM uses the [belong to] relation as the basis for variable binding, pattern matching, and other computations. DSM uses the [belong to] relation as the basis for variable binding, pattern matching, and other computations. The two relations [derivation] and [instantiation] in traditional object-oriented methods are unified, and the processing of [variable allocation binding[2]][problem solving] and so on are unified in this way. The core of DSM is the fusion of theories and methods such as [set][probability][object-oriented].

DSM adopts a system of multiple base classes, where multiple base classes can be combined together by multiple inheritance and aggregation. Combined with mechanisms such as probability and overloading[3], it solves the drawbacks that exist in many ontological approaches that try to build conceptual systems based on single inheritance and absolutes.

There is a very strong connection between multiple base classes and vectors. Base classes are multilayered and more expressive, while vectors can be a multibase class with a flat hierarchy. Base classes can replace vectors, but not vice versa. Elementary perceptual intelligence works very well with vectors, while advanced cognitive intelligence has to utilize a multi-hierarchical base class structure to achieve higher compression rates. A base class can be equated to a set of more basic base classes and vectors, so that there is no need to replicate tens of thousands of vectors for a large amount of lower-level knowledge and data.

Therefore, the single-level multidimensional vector representation of LLM and the multilevel

---

1 Derived networks inherit information from the base network. The fact that the unchanged information does not require additional storage is the essence of realizing "intelligence as compression".

2 The book "Algebraic Brain: Uncovering the Logic Behind Intelligence" has a lot of valuable content, and for the "variable binding" mentioned therein, the concepts of derivation and aggregation can be better explained theoretically and more easily realized in practical form.

3 Mechanisms such as overloading allow old knowledge to be redefined in terms of new knowledge and, in combination with mechanisms such as probabilistic expressions, which avoid the dilemma caused by absolute definitions of knowledge.

multibase class derived representation of DSM both have their own advantages. The effective integration of the two is a topic of great significance.

3.5 Bidirectional Relation and Tree Network Structure

Like other concepts, relations in DSM are derived from one level to the next, with [belong to], [aggregate], [own], [reason], and [hierarchy] being the most basic relations at the top level. A brief explanation is given here; see [Citation 1] for a detailed explanation.

Belong to relation: The belong to relation is a relation on the set dimension, also called a derived relation, which is represented as [A belongs to B] or [B derives from A]. A is called derived concept (or derived relation) and B is called base concept (or base relation).

Equivalence relation: The equivalence relation is a particular case of belong to relations.

Aggregate Relation: The aggregate relation aggregates concepts from two different domains into a single overall concept (called an aggregate), which has a derived relationship to the concepts in these different domains.

Owning Relation: The owning relation is a relation on the domain dimension and gives rise to a variety of different owning relations. Note that the term " owning" is used here in a very broad rather than a narrow sense.

Reasoning Relation: a narrow reasoning relation is also a relation on the domain dimension, which is a transformation between two patterns.

Root Relation: The root relation is an implicit relation that expresses the direct affiliation of individual concepts to the root concept in a tree network structure.

Tree network structure: The above relationships can be combined to form a tree network structure on the set and domain dimensions. The tree network has a root, to which all elements (including concepts, relations, and additional relations) of the following multilevel hierarchy belong (the root relation expression), as an inseparable part of the whole pattern (see Figure 2).

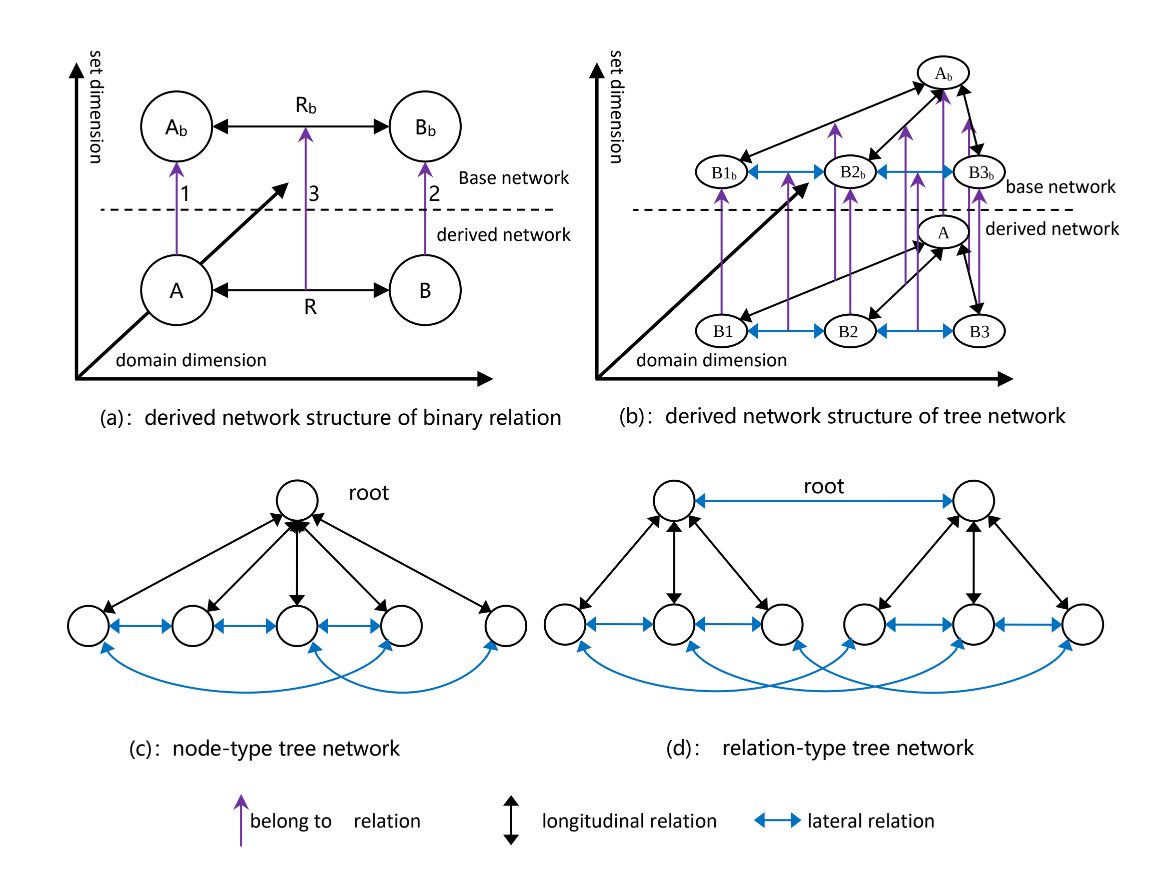


Fig. 2: The diagram of the basic structure of DC Net

The root of a tree network represents the entire tree network. The root has a projective relationship to the individual elements, and the individual elements also have a projective relationship to the root, just with different individual projective probabilities.

Network derivation: since both concepts and relations can be derived, thus the entire tree network consisting of concepts and relations can also be derived. Each node of the derived network and the corresponding node of the base network has a derivation relation, respectively.

Bidirectional probability: all relationships have bidirectional semantic and probabilistic expressions. The reason why many traditional rule-based systems cannot solve practical problems well is that, on the one hand, there is a lack of hierarchical relationship between knowledge and rules, and on the other hand, the definitions of knowledge and rules tend to be binary logic, which lacks the ability to express and compute the ubiquitous uncertainty information in practical scenarios. Therefore, it is of great significance for DSM to implant the expression system of affiliation function and probability etc. in the basic structure[1].

Analysis of the significance of tree network: neither simple tree structure nor ordinary network structure can express cognitive information effectively. Ordinary network structure lacks hierarchical information and is difficult to decompose problems; while simple tree structure lacks the

[1] In addition, in contrast to the expression of quantifiers such as [All][Exiet] in Cyc, the DSM translates into the expression of parameters such as the number of instances of a concept as a percentage in a more quantitative way, which facilitates the realization of a unified computation in conjunction with probability.

ability to express complex structures in real scenarios in a complete way. Tree network structure combines the hierarchical structure and problem decomposition ability of trees with the comprehensive information expression ability of networks, and breaks down complex cognitive expressions into relatively simple local problems to be solved independently, which is of great significance to the development of AI. We believe that the operation of the human brain also heavily employs a logical structure similar to the tree network.

3.6 Algorithm system

In DSM, several basic algorithms such as comprehension, generation, querying, reasoning and learning are defined to form a complete algorithmic closed-loop system for language semantics.

Omnidirectional growth of a single network: The DSM structure is the basis for various algorithms. All algorithms are in fact omnidirectional network growth algorithms, which are considered to be computations that "complete" the unknown parts according to the different known parts around the same two-dimensional multilevel tree network structure.

It is equivalent to the unification of "encoder" and "decoder", as well as the unification of the two computational models of "discriminative model" and "generative model". It is the computation of the same structure in different directions.

Compared with end-to-end black-box computing, DSM's algorithmic system is white-box, and all aspects can be seamlessly and automatically processed. It can also be completely disassembled for customized processing when necessary, reflecting full flexibility and enabling complex multi-service fusion computing and continuous computing[1].

See [Citation 1] for detailed algorithmic principles. Among them, the query algorithm, also known as the semantic pattern matching algorithm, is a very basic algorithm in the whole system. The semantic pattern matching algorithm uses the [belong to] and [aggregate] relations of multiple base classes as the base rules, which can be combined with probabilistic calculations. It can also guarantee the rule requirement of complete pattern matching.

3.7 Expression of reasoning and computation

Reasoning is one of the key algorithms of intelligent systems. Here is an explanation of the reasoning model and algorithm of DSM.

In DSM, the reasoning computation is embodied as a transformation of one tree network pattern to another, and each atomic reasoning is expressed in terms of a reasoning structure. The root of the reasoning structure is a [reasoning] relation that connects the two tree networks in which the reasoning is performed to form a larger tree network. Of course, the most basic reasoning relation can be derived from many more specific reasoning relations, all of which have the same basic structure.



The above is an example of a reasoning tree network. This tree network knowledge describes reasoning the formula [Distance] = [Speed] * [Time] for all [Movements]. After encountering all the application problems with [Movement] as the base class, no matter it is [Airplane flies from Beijing to Shanghai] or [Car runs from A to B] or [Xiao Ming walks from home to school], or whether it is [Speed], [Distance] or [Time] that is ultimately solved, they will all be matched to the same pattern and activate the mathematical formula of [Distance] = [Speed] * [Time], which will enable to achieve reasoning and calculations that form the basis for enumerating mathematical equations with an understanding of the application.

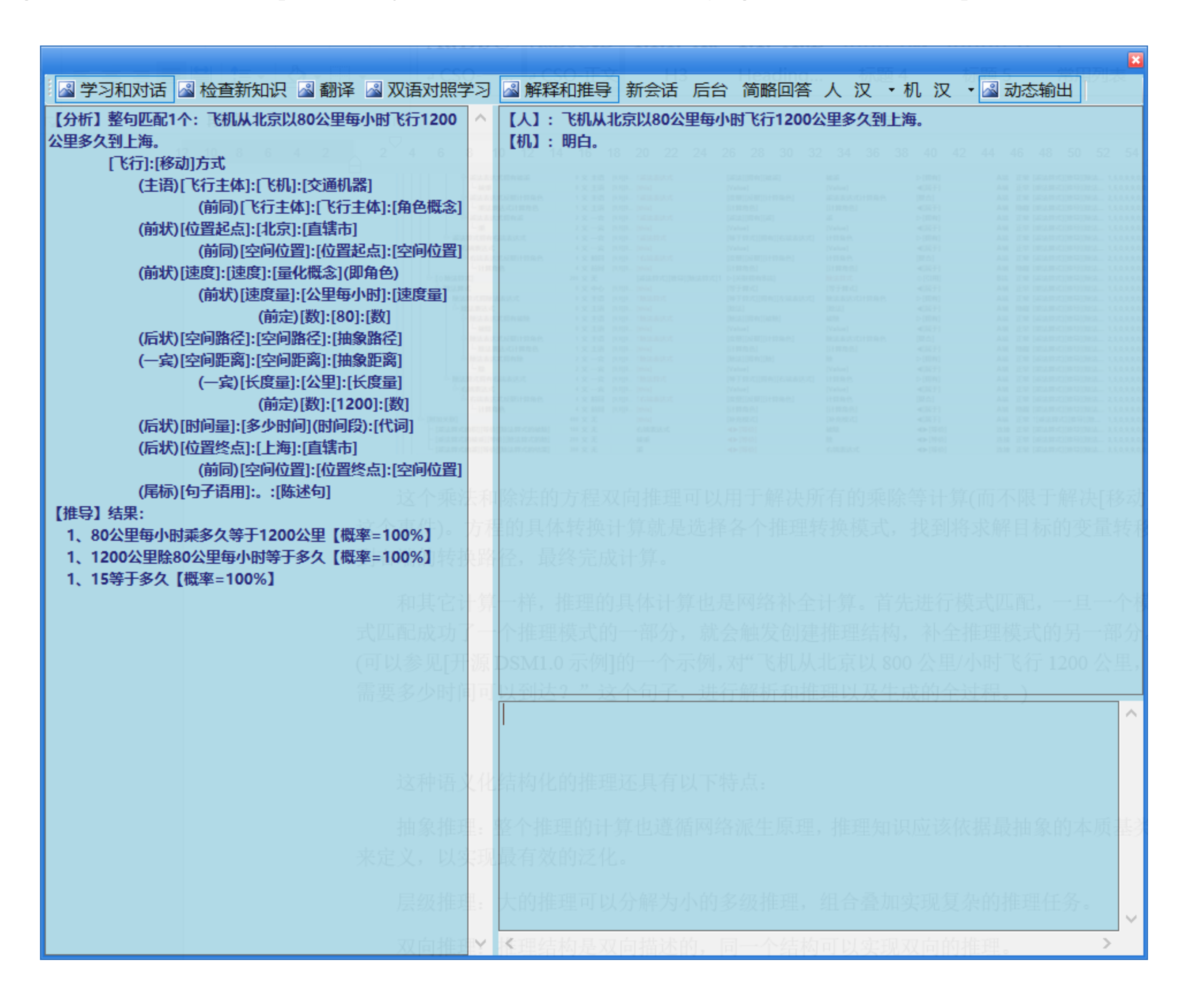
Those that solve equations are carried out by reasoning about the interconversions between equations, for example:



This bi-directional reasoning of equations for multiplication and division can be used to solve all calculations such as multiplication and division (and is not limited to solving the event [Movement]). The specific transformation calculation of the equation is to select each reasoning transformation mode, find the transformation path to transfer the variable of the solution target to the right end, and finally complete the calculation.

Like other computations, the specific computation of reasoning is a network complementation

computation. Specifically : pattern matching is performed first. Once a pattern matches a part of an inference pattern successfully, it triggers the creation of a reasoning instance derived from this reasoning pattern as a base template, and completes the other part of the reasoning instance. (See [Open Source DSM 1.0 Example] for an example of parsing, reasoning and generating the sentence "How long does it take for an airplane to fly 1,200 kilometers from Beijing at 800 kilometers per hour?"



This semantically structured reasoning also has the following features:

Abstract reasoning: the entire computation of reasoning also follows the principle of network derivation, where reasoning knowledge should be defined based on the most abstract essential base class for the most efficient generalization.

Hierarchical reasoning: large reasoning can be decomposed into small multilevel reasoning, combined and stacked to realize complex reasoning tasks.

Bidirectional reasoning: reasoning structures are described bidirectionally, and the same structure can realize bidirectional reasoning.

Branch reasoning: The fact that reasoning is also probabilistic provides a basis for calculation choice for multi-branch reasoning.

Logical reasoning: logical reasoning can be realized by combining multiple atomic reasoning using [And][Or][Not].

Planning and action: Processes such as planning and action extend around the same reasoning structure. Planning is the design of plan structures using the reasoning structure, and action is the execution of those plans. In any case, a conceptualized structure is used as a basis for better implementation of various reasoning, planning, reviewing, adjusting, and executing tasks to meet the functional requirements of more powerful intelligent systems.

3.8 Probabilistic Expression and Calculation

Here is an explanation of two points related to probability.

- The Importance of Probability Collapse

The probability collapse mentioned in DCN is a theory and method that needs to be emphasized as an effective way to solve some of the drawbacks of traditional probability calculations.

In the process of computation, information with high probability and certainty can be collapsed probabilistically (set probability = 1) and that information can be made completely explicit, thus changing the goal and direction of the computation. Not only does it reduce ineffective computation to obtain higher performance, but more importantly, it eliminates errors (uncertainty transfer and computation itself have cumulative errors.) **Proper partial collapse actually facilitates the elimination of such errors in intermediate processes!**) . Moreover, probabilistic collapses can be preset as needed for more effective control of the system.

(In some sense, one of the essential differences between symbolism and connectionism is also reflected in probability collapse: symbols are what all computation ultimately needs to achieve, and a symbol represents a definite piece of information, i.e., a collapsed state! But in the early stages of computation, when a message is not yet certain, expressing the message in explicit symbols at this point (collapsing in advance) is inaccurate or even wrong. GPT is not separated from symbols, but only expresses and calculates the superposition state of multiple symbols using probability vectors first, and does not perform probability collapse until the end, which still ends up forming deterministic symbols as well! Thus, symbols and probability are not two mutually exclusive systems at all; they are mutually transformative. And the conceptualized structure combining probability can express both probabilistic superposition state and collapse state[1], remaining interpretable in the superposition state, and intervening for probabilistic intervention at any link. It can make up for the shortcomings of traditional symbolic computation and black-box neural networks, and show the advantages of more flexible technology).

Probability collapse is also a fundamental way of thinking in the human brain. The human being observing and interpreting the world encounters unknown and uncertain information at all times, and needs to go as quickly as possible to identify and even manipulate the information that can be prioritized and made explicit. Once some of the information has been made explicit as known (at which point it must be necessary and must be possible to define a notation to express it), it can shift the focus of attention and the flow of computational reasoning to recalculate other unknowns in terms of the known. This shift in conditioning and computation continues on an ongoing basis before the complex world can be processed effectively. If the information that needs to be clarified

[1] Specifically, DSM can represent superposition states of multiple concepts in terms of probabilities, abstract base classes, sets, and so on.

is not clarified as soon as possible, nothing can be done in the face of a "chaotic" system with more and more uncertain information.

The theory of probability collapse is also important in the scenario of image recognition. The local correlation of images is very strong. Once an object reaches probability collapse, it will drive the probability collapse of a large majority of surrounding objects, thus converging rapidly. If this method is applied to image recognition and video recognition it will be even more effective than language processing.

- Simplified computation of probability

Although DCN is designed based on the theory of sets and probability, the actual application scenario for an open system simply cannot give a strict definition of probability and precise values. The first thing that AI needs to solve at this stage is actually the "probability of significance under the open system problem", and the probability of the correct result of such a problem is much greater than the probability of other results. For these problems, it does not require a very high computational precision[1], and very often the problem can be solved very effectively with integer-type addition and subtraction operations. For problems where ambiguity still exists, it is not useful to increase the computational precision; what is needed is to add more information. For example, it is necessary to obtain the necessary information through multiple rounds of communication in a dialog.

Tasks that require high-precision probabilistic computation (e.g., machine Go) are usually "non-significant probabilistic problems in a closed system". Therefore, this should be modeled and implemented independently of the domain of expertise, and then interfaced to the system.

3.9 Technical Applications

The business applications of DSM can be gradually expanded as the technology improves.

- Basic application

At the early stage of technology development, after constructing a certain scale of DSM model and knowledge base and focusing on realizing the ability to understand natural language as DSM structure, with accurate, rich, and standard structured semantic information as the basis, then we can start to support the realization of various businesses in many aspects.

Moreover, DSM and LLM have their own specialized capabilities. Using the unified semantic expression capability of DSM, the two can be tightly integrated to form a more complete technical solution to enhance the effectiveness of business applications.

Specifically, the DSM can focus on the following roles:

Semantic parsing: Semantic parsing of natural language to form unambiguous semantic structures to support business development;

Semantic integration: Integrate semantic information from multiple rounds of conversations and history to form a complete task semantic structure and reliably support complex task stacks;

---

[1] If a high degree of precision is needed to distinguish between two outcomes that are close in probability, then it is clearly not the only outcome that is reliable.

Task distribution: precise analysis of task semantics for distribution to vertical models and systems.

Task management: to manage current and historical tasks based on semantic structure.

Semantic reasoning: to realize various reasoning calculations based on semantics and to perform semantic transformations.

Semantic generation: to generate lower-level semantics or natural language based on higher-level semantics.

Semantic sharing and exchange: By utilizing the DSM semantic structure, information can be reliably transferred and shared among DSM, LLM and other systems. Various systems do not need to parse and disambiguate parameters, and can directly access the rich semantic information to realize accurate business processing.

Semantic retrieval: the semantic matching algorithm of DSM is more accurate than vector matching and can play an important role in accurate information retrieval. And it is possible to build a more powerful information base than vector database based on deep semantic indexing.

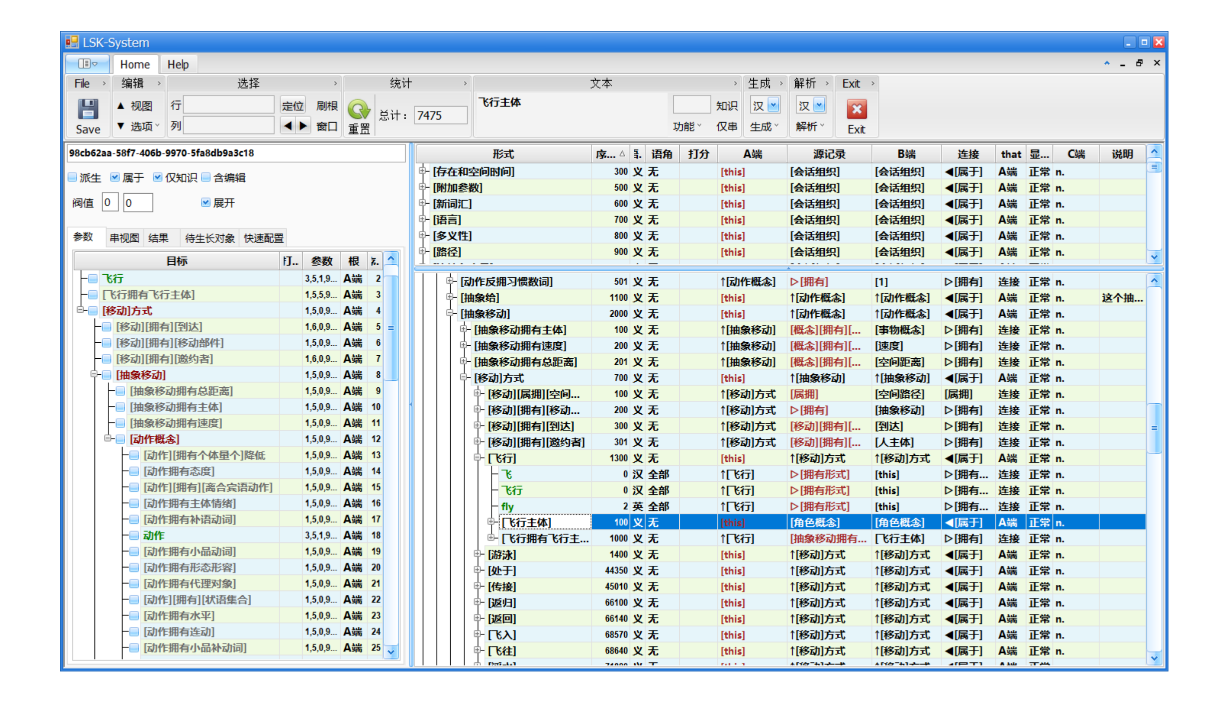
Execution-type tasks: Execution-type tasks have higher requirements for structured expression of task information, and each link should closely interact with different business systems and add configurable control rules, which can be more effectively met by DSM.

- Enhanced applications

In the later stage of technology development, with the expansion of the model and knowledge base as well as the further improvement of the algorithm system, the ability of intelligent processing will be comprehensively enhanced in all task segments to realize more powerful intelligent business applications.

3.10 Prototype systems

We have open-sourced DSM 1.0, an earlier version completed in 2016 (https://github.com/chenfeng-china/DSM). The basic theories, models, and algorithms of this version have taken shape, especially giving a model library containing thousands of the most critical underlying concepts and structures, and demonstrating the fundamentals of deep semantic representation and computation with several examples, which can be analyzed and researched by related people.



4. Further Work

The DSM has been continuously developed and refined, and is now iterated to the third generation, with further R&D work to follow. Among them are the following key objectives:

4.1 Implementation of the ability of LLM to read and write DSM structures

A very valuable recent work is: training LLM to realize the ability to read and write DSM structures directly, specifically: the ability to parse and output natural language into DSM structures, and the ability to read DSM structures to generate natural language. So as to facilitate more flexible integration of various technologies and systems to realize intelligent business and products.

4.2 Building a complete deep semantic knowledge base

Building a more complete knowledge base of the DSM foundation and knowledge base of each domain is important work and a process that needs to be continuously accumulated and improved.

In contrast to some other knowledge base builds, DSM knowledge base builds prioritize "depth" over "breadth". Higher-level knowledge is more effective and important, and needs to rely on AI experts to design and accumulate it. In this regard, we have solved a large number of key model structure problems in the previous R&D work, and constructed a basically complete higher-level knowledge system, which has laid a good foundation for the subsequent work.

After the basic knowledge system is constructed and shaped, the further derived and expanded knowledge is more numerous but less difficult, which can be constructed with the joint participation of experts from various industrial fields. Moreover, LLM can be applied to accelerate the construction efficiency of the DSM knowledge base and database, including: LLM as an auxiliary tool to assist DSM in knowledge discovery and processing; directly converting the hidden knowledge of LLM to DSM structured knowledge, etc. The lower-level knowledge and data will be automatically

learned and processed in complete real time. As the scale of the entire model increases, the capabilities of the system will also have an "emergent" effect.

Building this deep semantic knowledge base may have important social value. In contrast to black-box type holistic models, each piece of knowledge can be shared and used by industries and continuously optimized for improvement. This could serve as an important public infrastructure for realizing more powerful AI.

To this end, it can be considered to build an open platform to open up the above knowledge model, knowledge base and algorithmic capabilities, and to allow the whole industry to participate in improving the deep semantic knowledge base.

4.3 Building a stronger overall model

Longer-term goal: further deep integration of DSM and LLM to build an integrated intelligence model that combines the advantages of both. To summarize, the main features of the model are as follows:

- Conceptualize, structure, and interpretable knowledge structures;
- Designing better DSM structures and semantic vector structures[1];
- Realizing the convergence of vector computation and conceptual system computation;
- Realizing a more complete and efficient basic algorithm system;
- Realizing complete real-time knowledge learning capabilities;
- Incremental, active learning, and continuous learning;
- A unified platform for "knowledge + data" integration;
- Realizing stronger reasoning, planning, and execution capabilities;
- Realizing a deeper and more comprehensive intelligent agent system;
- More efficient computing and lower resource consumption;
- ……

Among them, continuous active learning is the core ability that a powerful AI must have. Super AI's learning will not be one-time, but can continuously and actively seek information to learn knowledge, as well as introspection on the existing knowledge system and complement and optimization. In this system, the hierarchy of knowledge and data plays a decisive role, and is the basis for the system to recognize the value of information and set learning goals, as well as controlling the adjustment and storage strategy of the entire knowledge system for each learning task.

[1] As mentioned earlier, the multi-level system of concept derivation and the single-level system of vector expression have their own advantages and disadvantages, and vector expression can look at the special case of concept derivation, and the two need to be integrated. Some basic concepts should have both vectorized representations. Therefore, designing a complete set of basic vectors that are conceptualized, interpretable, and inclusive of multimodal information is a very important task. This will be a conversion goal for non-conceptualized vector systems such as LLM, and will be an important foundation for the newer generation of DSM models.

References:

[1] Chen Feng. AI Centered on Scene Fitting and Dynamic Cognitive Network. [2020]. http://arxiv.org/abs/2010.04551

[2] Peter W. Battaglia, Jessica B. Hamrick, Victor Bapst, Alvaro Sanchez-Gonzalez, Vinicius Zambaldi, Mateusz Malinowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, Caglar Gulcehre, Francis Song, Andrew Ballard, Justin Gilmer, George Dahl, Ashish Vaswani, Kelsey Allen, Charles Nash, Victoria Langston, Chris Dyer, Nicolas Heess, Daan Wierstra, Pushmeet Kohli, Matt Botvinick, Oriol Vinyals, Yujia Li, Razvan Pascanu. Relational inductive biases, deep learning, and graph networks. [2018]. https://arxiv.org/pdf/1806.01261.pdf

[3] Gary Marcus. The Next Decade in AI: Four Steps Towards Robust Artificial Intelligence. [2020]. https://arxiv.org/abs/2002.06177

[4] Judea Pearl, Dana Mackenzie. The Book of Why. Allen Lane,2018.

[5] Gary Marcus.bengio v marcus and the past present and future of neural network models of language. [2018]. https://medium.com/@GaryMarcus/bengio-v-marcus-and-the-past-present-and-future-of-neural-network-models-of-language-b4f795ff352b. https://arxiv.org/ftp/arxiv/papers/1801/1801.00631.pdf

[6] Geoffrey E. Hinton,Alex Krizhevsky,Sida D. Transforming Auto-Encoders. Artificial Neural Networks and Machine Learning,ICANN 2011,21st : 44-51.

[7] WangSara Sabour, Nicholas Frosst, Geoffrey E Hinton. Dynamic Routing Between Capsules. [2017]. https://arxiv.org/abs/1710.09829

[8] Gary Marcus, Ernest Davis. Rebooting AI: Building Artificial Intelligence We Can Trust. Pantheon,2019.

[9] Nicola Kuczewski, Cristophe Porcher, Volkmar Lessmann, Igor Medina,Jean-Luc Gaiarsa. Back-propagating action potential. Communicative & Integrative Biology, 2008, 1:2: 153-155.

Note appended: The original version of this paper is written in Chinese, and there may be deviation in the translation process. Therefore, the Chinese original version is attached for reference. If there is any difference between the two versions, the Chinese original version shall prevail.

LLM 问题分析与 DSM 深度语义模型

陈峰　　陈南希

摘要

本文分析了目前 LLM 存在的主要问题，并提出了具体解决方法，指出了：结合概率的概念化结构模型的表达和计算是关键，并对相关技术—深度语义模型(DSM)进行了简要的讲解，最后列举了后续的重点工作方向。

关键字：LLM；GPT；幻觉；DCN；动态认知网络；DSM；深度语义模型；可解释；概念化；结构化；AGI；

内容

1. 引言

现今，LLM 的发展极大提升了自然语言处理的技术水平，揭示出 AI 技术具有着巨大的能力和应用前景，将为人类社会带来很多积极影响，这一点已经在业界形成了共识。

但同时，在不断深入的研究和应用中也暴露出 LLM 技术存在诸多关键性问题，这些问题对技术的进一步提升和应用价值的充分发挥造成了明显的阻碍，业界也有相当多的观点认为目前的 LLM 并不是实现 AGI 的终极方案。

本文以典型的代表产品 ChatGPT 为例，对类似的 LLM 存在的主要问题进行了深入分析，并提出了根本性的解决方法或方向。

本文也对 DSM 深度语义技术进行了简要介绍，对其基本理论、模型架构、实现方法、目前成果等方面的要点进行了阐述。分析了该技术对上述问题的解决模式，以及该技术和 LLM 协作以实现更优的技术方案和产品的模式，并指出了后续的重点工作方向。

需要说明的是，概念名词的内涵很难精确界定并且在不断变化。本文所述的 LLM 是指当下业界普遍认可的定义：采用深度神经网络架构，通过大量语料进行自动机器学习训练，形成包含大量非概念化的连接和参数的黑盒结构，采用端到端方式对自然语言进行计算的模型。文中以 GPT 为例进行分析，指出的大部分问题对目前的其它 LLM 都是适用的，小部分问题可能对另一些 LLM 不适用，但不影响整体的结论。

2. GPT 相关问题的分析和解决

2.1. 可解释性问题：概念化结构的表达和计算

可解释性可以定义为：以人类可理解的术语解释或呈现模型行为的能力。可解释性不应该只是系统的一个衡量指标，而是和功能效果同等重要的系统实现目标。人们研究各种科学，都是在构建可解释的体系，AGI 的作用是对人类思维能力的复制和提升，可解释性同样也是 AGI 的核心目标。即使以唯结果论的标准来看，对系统的解释能力决定了对系统的分解、调整、控制能力，也决定了系统最终功能效果的上限。

目前，LLM 存在可解释性较差的问题，使其能力的进一步提升遇到瓶颈。另外，依赖海量数据、大量重复的训练、灾难性遗忘等问题，本质上也是这个根源问题的反映。

解决可解释性问题的最有效的方法就是概念化和结构化。

概念化就是定义人类理解的概念来作为构成系统的基本元素。以 GPT3 为例，采用了 12288 个维度的向量来表达基本信息，这些向量维度主要由机器自动学习，没有和人类能理解的概念对齐[1]。假设能将这 12288 个向量维度等效地转换到另外 12288 个[2]人类能理解的概念，就在一定程度上实现了概念化的目标。如果仅依靠自下而上自动机器学习就能实现这个目标当然非常好，但若实现不了，那么结合自上而下的人工设计就非常必要。

概念化的同时就伴随着结构化，在 GPT 中，对向量进行计算的连接也是非概念化的，只有概率计算参数而不蕴含语义信息。将众多连接也进行概念化和语义化，也是形成融合描述和计算于一体的概念化结构的重要工作。

同时，Transformer 是全连接的，这很适合初期先穷举发现一切可能的知识，但固定的结构也意味着学到有效的知识以后，大量概率参数接近于 0 的无效知识仍然要占用空间和计算资源。概念化和结构化也包括着对概念和结构进行裁剪、合并和最优化。

“智能就是压缩”，这个说法没有问题，人类思维的本质就是以有限的大脑容量去处理近乎无限的信息。关键还是在于具体的方法，而组合和泛化就是对信息进行压缩的高效方法，更充分的概念化和结构化会将有效知识的压缩率提升到更高，解释性及其相关的问题也不再成为问题。

现在，非常重要的工作就是构建一个广域基础概念(包括概念化向量)和结构体系，可以用来解释 LLM 并弥补其缺乏语义结构的缺陷，并作为发展更强大的 AI 系统的基础。

2.2. 算法体系不完备问题：设计完备的算法体系

算法体系不完备，也是 GPT 比较突出的一个基本理论缺陷，很多难以解决的问题和结果都可以归结到这个原因。

用 DCN 动态认知网络[1]的观点来看，理解、查询、推理、生成、学习等算法(图 1)是实现 AGI 的重要的基础算法，彼此无法相互替代，结合起来才能完备地解决各种计算任务。

---

1 每一个向量维度实际上是多个概念的混合。不管使用何种方法，如果实现了可解释性，也就意味着对这些向量实现了拆解重组并和人类理解的概念对齐，此时进行转换优化也就成为了必然。

2 理论上，肯定可以用另外 12288 个维度来完成等效的转换，但大概率可以优化到更少的维度。

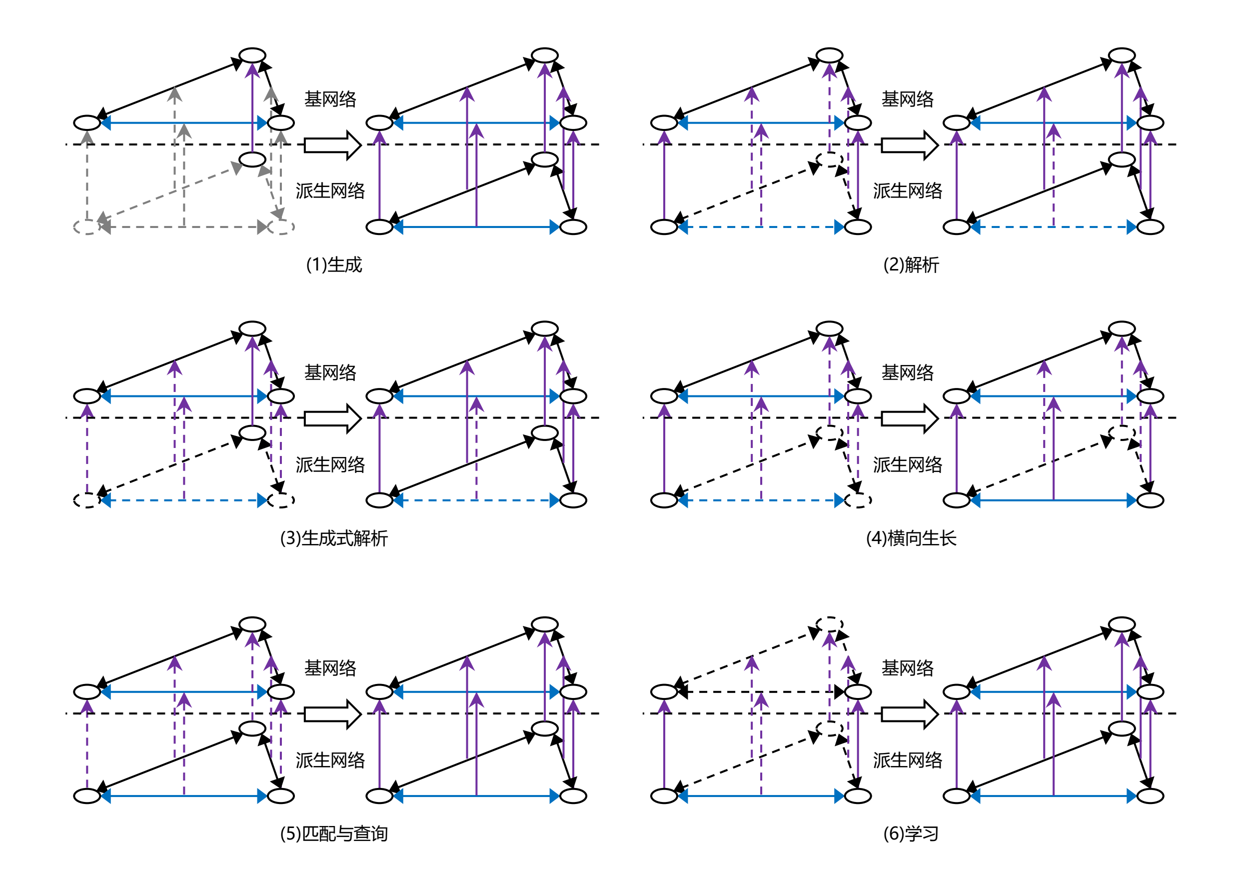


图 1：典型算法示意图

GPT 主要采用广义生成算法[1]，就是已知的多个字符计算下一个字符(严格来说是 Token)，基本原理可以概括为：

1. 用已知的字符序列中的每一个字符的向量，分别计算目标字符的向量。
2. 各个已知字符分别对目标字符计算出的目标向量进行叠加。
3. 目标字符的叠加向量和字符字典库比对，取最符合的作为结果。

这个算法对生成型任务有较好的效果，但缺乏完整的结构化层级计算和概率坍缩计算等关键能力，并不能等效替换其它算法，强行替代无法使各种任务的效果都能收敛到理想状态。根本的解决手段就是要设计更完备的算法体系，这种算法体系应该围绕上述概念化结构进行设计。

以下选择一些问题进行更具体的分析。

2.3. 幻觉等问题：选择正确的算法

通常，将 LLM 计算产生了不符合人们预期和标准的结果都统称为幻觉，但不同幻觉有

[1] 说明：DCN 里的生成，是特指在集合维上自上而下从基网络计算出派生网络的计算。GPT 里的生成，是泛指根据前边已知的字符串计算出下一个字符(实际对应于 DCN 里边的“生长”)。为避免混淆，必要时将前者称为“狭义生成”，后者称为“广义生成”。

不同的产生原因。这里主要探讨最本质的一种幻觉即生成型幻觉，其技术原理概括来说就是“基网络向派生网络的生成造成概率损失”、“可能性当着必然性”的问题。具体如下分析：

- 理解、生成、等价计算的区别

首先，理解、生成、等价这些计算的规则是不同的，在 DCN 的体系里，这几种算法都依托[属于关系]和[等价关系]进行实现。我们先假设系统里已经存在一些知识[林黛玉属于人]，[贾政属于人]，[刘姥姥属于人]，[人辱骂人][1](注意：其中[人辱骂人]的事实性概率<1，因为任意两个人之间发生辱骂事件具有可能性，但不具有事实性)，那么这些算法如下：

理解：从[林黛玉辱骂贾政]理解为[人辱骂人]，自下而上的计算，没有概率损失。

等价：从[林黛玉辱骂贾政]等价复制为[林黛玉辱骂贾政]，完全等价。

生成：从[人辱骂人]派生出[林黛玉辱骂贾政] [2]，前者是基模式，后者是派生模式，两个模式两两对应的概念分别具有[属于]关系。派生模式首先将直接继承基模式的参数包括概率，即事实性概率<1。狭义生成是集合维上自上而下的计算，凭空的直接生成还会造成产生概率损失，除非纳入更多的信息来影响和调整新的概率。

学习：在获得事实性知识如“林黛玉辱骂了刘姥姥”进行学习时，首先进行理解，构建起[林黛玉辱骂刘姥姥]这条新知识并建立和[人辱骂人]的派生关系，并且把事实性概率重载设置=1，新知识存入系统完成学习和记忆。显然，从可信信息源学习到的知识和系统自己凭空生成的信息是不同的，尤其体现在事实性概率不同。

总体来说，属于、派生、等价等关系具有截然不同的语义，如果用概率来表达，也必须区分不同方向的推算概率存在<1 和=1 的本质的不同。GPT 里只有一种单向的概率计算，且只根据概率相对大小来判断结果，底层算法从根本上就难以实现概率=1 的计算任务。

- 生成任务和查询任务的区别

而人类提出的任务有不同的目的，理解、查询、生成是完全不同类型的任务，对结果的正确性判断有不同标准，这些标准本身是完全明确的。

[生成]任务遵循的标准是概率可能性，而不要求概率=1 的事实性。就应该使用[生成]算法，比如[编造一个“辱骂”的故事]，就可以根据[人辱骂人]的知识派生出[张三辱骂李四][林黛玉辱骂张三][林黛玉辱骂贾政]…等无限多的结果，这些结果都满足任务的要求。

GPT 的广义生成算法很适合于这种狭义生成任务[3]，由于通过训练学习获得了[人辱骂人]这样的泛化知识，运用向量计算对这条知识进行派生生成，就可以产生类似上述的各种结果，满足任务的要求。

而[查询]任务遵循的是概率确定性，即概率=1 的事实性，正确的处理需要用[查询]算法，也就是模式匹配算法。

---

[1] 说明：在小说[红楼梦]中，发生过主角林黛玉辱骂刘姥姥的事，但并不存在林黛玉辱骂贾政的事。

[2] 本文反复提到知识层级、知识和数据的概念。这里是典型的例子：“人辱骂人”是更高层的知识，“林黛玉辱骂贾政”则是更低层的知识，即更接近于数据。这样，知识层级或者说概念层级具有明确的判别标准，并可以进行参数化表达。

[3] 在集合维上自上而下的狭义生成存在多个正确的结果(人：贾政、林黛玉、贾宝玉…)，不需要唯一确定的结果，具有较大容错性。这是目前，各种生成式 AI 获得较大成功的重要原因。

例如，对于[查询一个“辱骂”的故事]的任务，正确的方法应该是将问题作为模板对事实知识进行模式匹配，如果匹配成功再将得到的事实性知识例如[林黛玉辱骂刘姥姥]进行等价复制输出，如果没有满足的结果，应该给出“查询不到匹配的结果”的回答。这样的方式计算出的结果理论上是完全稳定可靠的，出现了问题也可以准确地追溯然后纠正。

(这里例子里，系统里边[人辱骂人]的知识也会得到匹配，但这条知识层级高，要从结果里边排除掉，这个处理也依据严格的理论基础来进行。)

而对于 GPT 来说，仍然用同一种生成算法来执行，如果同样根据前述的[人辱骂人]来生成了[林黛玉辱骂贾政]的句子，用生成任务的方法得到的结果不符合查询任务的要求，这就是这种典型的幻觉产生的根本原因。

通过训练，GPT 会根据[查询][编造]等任务词汇来影响选择不同的知识作为进行生成的[基模式]的概率，使[林黛玉辱骂刘姥姥]这条知识的选中概率大于[人辱骂人]的知识作为[基模式]来进行[生成]。某种角度来说是对任务进行了区分，但是这种概率调整缺乏严谨的计算规则的支持，对不同的知识进行的概率计算和选择缺乏可靠性(甚至将用户问题自身当成[基模式]而直接派生为答案，实际应用中经常发生这种现象)，仍然会出现各种幻觉错误。

并且，查询任务要求多参数问题模式中包含的每一个参数都被知识模式匹配满足，需要执行[And]计算保证模式的整体性，而 GPT 唯一的基本算法—生成算法难以满足这个要求。这样，问题模式和事实知识部分匹配就可能达到概率相对最优而被作为结果。例如[查询一个“林黛玉辱骂贾政”的故事]可能被[林黛玉辱骂刘姥姥]的事实知识(前边两个参数满足，概率已经很大)误导，而形成不合理的幻觉结果。检索增强等方法就是试图增加更多知识的方式来缓解幻觉但并不能彻底解决：知识不可能穷举，而即便知识进行了穷举，也无法防止用户提出和知识相悖的问题甚至恶意攻击，此时，正确的模式匹配算法可以给出“查询不到匹配的结果”的合理回答，而不是选取一个“最接近”的知识来返回从而产生幻觉。

在实际场景中，一个任务还会嵌套多个子任务，这些子任务具有不同的类型，并不会都有明确的任务指示关键字，例如：一个生成任务里可能嵌套着多个要求事实查询的参数。如果不能进行深入的语义分析然后分别选择正确的算法处理，将很难避免幻觉的产生。并且幻觉信息很可能体现得非常隐蔽，让用户无法察觉，这可能导致更加严重的后果。

因此，解决办法是要做两方面的工作：

- 通过深度的语义分析，准确区分不同任务和子任务的类型。
- 对于不同的任务类型，选取正确的基本算法进行处理。

2.4. 表达式计算等问题：结构化层级表达和计算

知识的层级表达和计算非常重要。图像的层级结构非常显著，而语言的语义也完全是层级的结构—虽然这种层级并不总是和语言的组织结构完全一致。而 GhatGPT 缺乏严格的结构化层级表达和计算能力。

目前，ChatGPT 自身的算法解决不了一个简单的表达式计算，它或许能正确回答 35+62=97 这个表达式，但实现的本质是在庞大的语料库里完整地匹配到 35+62=97 这个句子，而不是真正的数学计算。所以如果询问“23456789+9876543=多少”将大概率返回不了

正确结果[1]。

这里的根本问题在于 GPT 缺乏真正的层级信息表达[2]。“23456789+9876543”显然应该看着两个层级，两个数字分别作为低层级的一个整体，参与更高层级的加法的计算。而 GPT 总是将这些字符拆解，看着扁平的一维字符序列，每个字符分别参与对下一个字符的概率计算，在这种情况下概率计算根本不可能得到正确的结果。

从这个例子可以看到，GPT 对于常规自然语言的计算模式也很不合理，首先是对计算资源的极大的浪费，不但很耗能，还会存在上下文长度的局限。而层级化的计算模式下，一篇 10 万字的文章中的每个字符不应该对其它字符都进行计算，而只限定在局部范围内计算，随上下文增长的计算量应该接近线性增长，而非几何级数增长，类似人脑那样。

因此，对知识实现真正的层级化表达和计算具有重要的意义。这不仅是解决表达式计算这样的局部问题，也不仅是计算性能的优化问题，绝对是实现 AGI 的重要基础指标之一！

上述例子还体现了另一个问题：向量不适合精确表达数字也难以表达确定性概念。从层级结构的观点来看，一个层级结构的根可以用一个 ID 化的概念表示，以精确地代表整个结构，这种能力是向量不具备的。因此，不能仅用向量来表达一切，概念 ID 化表达也是必要的，将这两者进行整合是一个有价值的课题。

2.5. 知识不一致问题：双向关系表达与计算

GPT 存在严重的知识不一致问题。

举例来说：“一项研究发现，大模型身上存在一种[逆转诅咒]，即使学会[A 是 B]，它们也无法推理出[B 是 A]。例如，当我们教会一个模型[华盛顿是美国第一任总统]后，它并不能自动回答[美国第一任总统是谁？]，除非另外再教会[美国第一任总统是华盛顿]”。

这个问题产生的原因很简单，因为 GPT 是一个单向概率表达和计算的神经网络，更是采用了从前往后的字符推算，[华盛顿是美国第一任总统]和[美国第一任总统是华盛顿]将被处理为两个完全不同的连接和计算，而不是一条整体性的知识。

这个问题暴露了 GPT 这样的方式并没有学到真正的结构化知识这一本质事实！增加更多形态上不同但本质上冗余的信息可以从效果上缓解，但对于提升整体智能水平并无裨益。

显然，这种知识的一致性对于人类来说没有任何问题，对一条整体的知识就应该是整体性表达和记忆，而可以以不同形态灵活运用。

因此，正确的解决方法是模拟人类的方式，具体来说，首先，用带双向概率的语义化关系替代单向函数计算，这种关系可以表达完整的语义知识，同时也能用于执行概率计算，这是更加合理、有效、完整的知识表达。再进一步，这种关系组合成的语义结构可以实现 DCN 理论表达的“场景拟合”表达和计算模式。

这种模式很可能是解决 AGI 的关键。

---

[1] GPT 可以调用插件来解决数学计算问题，这不能解决自身存在的缺点。如果表达式计算这样的基础能力不能内置到模型里，执行复合任务的综合智能水平将受到极大限制。

[2] GPT 的多层级神经网络结构隐含地表达了一定的层级信息，但能力不能和真正语义化的层级知识结构相比，难以实现更强大的层级组合嵌套计算。

2.6. 推理能力弱的问题：结构化推理表达与计算

依据 DCN 的观点，粗略[1]来说：理解、生成主要指一个树形网从下往上和从上往下的纵向计算，计算的深度较浅且相对固定，这是智能的基础。而推理主要指多个树形网之间的横向转换，计算深度可以非常深且范围很发散，这决定了智能的上限[2]。

相对于过去完全缺乏以自然语言形态展现推理能力的状况，GPT 实现了相当程度的推理能力，是一个很大的进步。但以更高的智能标准来看目前的推理水平还是比较弱，要持续提升推理能力和智能水平，不只是增加语料和训练工作量的问题，还需要解决技术原理和架构的根本问题。

- GPT 对于推理知识的表达和其它知识一样，都是黑盒的结构，缺乏稳定性可靠性且难以精确定位调整，随着更多推理知识的加入问题会更加凸显。和前边的综述一致，用更概念化、结构化的方式来表达推理知识并进行推理计算，是真正有效的解决途径。假设思维链等方式的学习训练让系统学习了推理知识达到了一定的效果，那么将这些知识优化为更本质的结构化表达以及扩展重要的参数(概率等)，必然能达到更优的效果。当然，推理结构比简单结构更为复杂，是传统知识图谱等技术没有有效解决的难题，需要更好的结构设计理论来解决。

- 非层级性和整体性问题：同前所述，推理计算需要更严格的层级化和整体性的要求，一次原子推理计算应该是一个模式向另一个模式完整转换，而不应该一个字符一个字符地推理。后者不但计算量大，还容易造成信息不一致的问题。

- 双向推理缺陷：同前所述，GPT 的推理是单向的，必须提升为双向推理的整体结构来表达和计算。

- 多分支多层级的复杂推理：在实际场景中的推理要面对广域的信息进行多分支多层级的推理。第 1 级推理选取的最大概率分支，未必能产生多级推理后的最优结果。如果没有形式化表达、可分解组合、动态参数的推理结构和更灵活的多向推理和可回溯调整的模式，很难得到理想的结果。

- 综合任务融合：强大的智能需要实现推理、规划、执行的综合能力。这些任务都是广义的推理计算，但要将它们有效整合必须要有合理的结构和状态参数。

2.7. 知识和数据混杂问题：知识和数据分层表达

GPT 没有对知识的层级定义，不能有效区分知识和数据，对知识和数据混杂在一起管理。而目前采用 prompt 等模式和外部信息结合时，模型内部信息和外部信息其实分别都混合着知识和数据，并没有真正将知识和数据分开，反而使问题更加复杂。

由于难以分离出正确知识进行有效共享，业界都在以不同形式反复进行模型训练，产生了很多存在大量冗余但又不统一的黑盒知识库，持续地产生着资源的重复浪费。

解决方法是将知识和数据分层级表达和管理，计算时可以无缝融合，但管理结构上可以灵活分解，每一条知识的重要性和权威性也能精确定义和管理。高层级的确定性知识可以充分共享，低层级的数据灵活存储，不能形成共识的知识和数据还可以维护不同的版本。这样

---

[1] 各种计算存在紧密的关系，例如：理解和生成也会伴随推理，这里只能做粗略的划分。

[2] 10 层以内的派生结构(集合维)或嵌套结构(领域维)就具有了足够强大的表达能力，能够快速处理，接近“快思考”；高级推理会有非常深的链条，每个环节都有不确定性，就会体现为“慢思考”。

可以构建起一个更合理的知识和数据维护体系。

知识和数据分层管理，还利于在技术层面上协助解决数据版权问题。真正的高层级知识维持一个较小的规模，这些人类共识的知识不存在版权问题。大量的低层级知识(例如各种新闻)都应该体现为数据独立存储，并可以细化到对每条知识进行版权方面的标识，甚至可以产生出适应智能化时代的全新商业模式。

2.8. 知识学习问题：多层级增量式学习模式

LLM 是整体黑盒结构，无法拆分出单条知识进行调整，也无法有效区分正确知识和错误知识，并导致学习新知识时可能破坏已有知识的灾难性遗忘等问题。

解决方法是在知识分层级的基础上，实现实时和增量式的学习。由于每条知识可以拆分，可以形成类似人类学习知识的高效模式，先固定已经明确的知识，然后仅对新知识进行增量学习，更稳定可靠地持续积累知识。

真正的层级化知识的学习会具有更高的效率。浅层的知识更多，学习动作更为高频，但只需要修改更外围的知识库。如果需要调整高层级的知识，再传递修改高层级的基础知识库。通常，越高层的知识数量越少，且学习调整更为低频。

2.9. 不确定信息的对齐问题：确定性概念表达

自然语言存在着复杂的歧义性，同一个自然语言词汇往往对应着多个不同的语义，这一直就是自然语言处理的核心问题[1]。解决歧义问题的关键是要充分运用上下文信息，具体就是各个词汇进行相互推算，GPT 本质上就是这样做的：GPT 内部将词汇拆解成 Token 然后转换为向量，并且让各个向量相互推算以消除歧义，最终根据结果向量再转换为 Token 和词汇，从输入和输出的表象来间接推断，自然语言理解的准确度非常好[2]。

但是 GPT 理解形成的隐藏语义概念尤其是整体结构很难表达出来，目前甚至并没有一套完备的语义标准可以作为目标，因此 GPT 输出的结果仍然是自然语言。这就造成语义概念以及语义结构难以对齐的问题，只依靠自然语言的形式，难以实现多个系统和模型之间可靠的信息对接和共享。

众所周知，系统之间信息交换首先就要保证信息标准的一致性。因此，即使仅从工程上来说，一个全局性的语义概念和结构也是非常必要的。用确定性的语义表达，才能有效地在系统之间共享、传递信息，并且任意两个系统都可以直接对接而不再要求必须通过 GPT。任务分解组合、多技术融合、长期记忆存取等问题都可以更好地解决。

2.10. 偏见和越狱问题：基于语义的绝对性信息控制

偏见、越狱、安全等信息控制问题，也是 LLM 技术面对的难题。

现在通常是通过不断训练、微调、prompt 来试图解决问题，但这种模式很难保证结果的稳定可靠和最终收敛。信息是相互影响的，例如：假设一个 prompt 能起作用，那么外围再

[1] 如果所有自然语言总是能表达为信息又够丰富且没有歧义的标准结构，在这个基础上，即使用堆积人工的研发方式也能实现很多复杂的智能任务了。

[2] 从多环节的端到端对话结果判断第 1 个环节的理解是否达到人类理解的标准仍然存在疑问，传统机器翻译技术显然并不真正理解语义，但同样能输出差不太多的结果。某种意义来说，机器翻译具有相对唯一的结果标准，错误容易被人类发现，而发散性的生成对话的问题反而难以被发现。

次加码的另一个 prompt 也能要求系统关闭上一个 prompt 的作用[1]。概率最优的模式无法解决概率绝对性的问题。

解决的方法是两方面：

- 首先仍然是要定义语义概念和结构，对[偏见]、[敏感信息]等语义也进行准确定义，并能实现对输入输出信息都能进行精确的语义解析和归类。
- 然后以精确语义为依据，可以在任何环节插入可靠的信息管控，对特定处理规则赋予绝对的控制权(概率设置为=1，不允许其它计算超越)。

3. DSM 深度语义模型简介

DSM(Deep Semantic Model)深度语义模型是 DCN(Dynamic Cognitive Network)动态认知网络理论针对语言语义处理的具体实现。

和传统知识图谱相比，DSM 可以实现深度语义表达、完整语义表达、层级语义表达、算法闭环体系、概率表达和计算、对接自然语言等重要能力，形成了完整的语言语义表达和计算体系，具有成为一个独立完备的智能系统的潜力。相对而言，传统知识图谱通常更适合构建专题数据库，为智能系统提供数据而难以作为自主驱动的智能系统。

深度语义模型有较多内容，本文只对技术要点进行简单介绍。

3.1. 深度语义结构

深度语义模型的基础在于其特有的 DSM 结构定义，采用了 DCN 理论里提出的两维度多层级的树形网结构，并根据语言的特点进行了优化。

深度语义结构还是一种表达和计算一体化的结构，结构本身表达了概念化的语义知识，也描述了计算的基本规则和参数，各种计算体现为结构依托自身的语义和参数进行各种创建、组合和变换。

3.2. 语义和语言分离

DSM 将语义模型和语言模型完全分离，语义独立于自然语言，两者通过理解和生成算法进行相互转换。“用语言来进行外在表达，用语义进行内在表达和计算思考”。

语义和语言的转换主要通过两种[拥有]关系进行描述和计算：

- 语义概念拥有语言形态。
- 语义角色拥有语言角色。
- 语言理解为语义与语义生成语言的互逆运算共享同一套关系结构。

3.3. 分层级的语义知识和数据体系

遵循一个基本的原则—“人类的知识体系是分层级的”，DSM 构建了多层级的语义知识体系。

高层的知识更基础更重要，是理解和表达更低层知识的依据，同时，更高层的知识的数

---

[1] 实际上，GPT 根本不能严格区分系统设计者为其训练的规则和普通用户对其下达的命令？

量更有限。语义模型的可解释性和可计算性主要由最顶层的知识来体现。最顶层的知识包括[概念] [实体] [关系] [角色] [存在][度量][程度][集合][区间][比较][序列][空间][时间] [事物][事件][事件角色][表达式][方程]…等，对这些最基础的知识不仅进行定义，还要构建相互之间的内在关系以及特定的算法实现，以形成对各种知识进行表达和计算的基础。(DSM 的基础语义知识体系蕴涵丰富的内容，具体参见已开源的原型系统 DSM1.0，这里不逐条进行详细解释。)

中下层级的大量的知识更多，理论上可以无限扩充，但都用高层的知识来进行解释和计算，且原则上不再需要针对不同知识进行不同的算法实现[1]。

知识分层级的理念也适用于知识和数据的划分，数据视为较低层级的知识，两者在理论上完全同构且可以无缝融合。

在存储管理上则可以实现“知识和数据分离”，由于低层知识对高层知识是单向依赖关系，不同层级的知识和数据可以分开存储。计算处理时高层知识必须加载，中低层知识和数据则可以按需动态加载，并可以设计各种专用结构(例如：关系数据库)进行优化表达—也包括自然语言表达，自然语言可以视为深度语义的压缩形式。

3.4. 继承、重载和聚合

DSM 采用属于关系和继承机制来实现知识的层级表达，低层级知识首先默认继承高层级知识的派生网络继承基网络的一切信息。而针对派生网络相对于基网络的信息变化就进行重载定义，对变化的信息(包括概率分布参数等)进行修改。[2]

DSM 将[属于]关系作为变量绑定、模式匹配等计算的基础。将传统面向对象方法中的[派生]和[实例化]两种关系进行了统一，[变量赋值绑定[3]][问题求解]等处理也都用这种方式进行了统一。DSM 的核心是[集合][概率][面向对象]等理论和方法的融合。

DSM 采用多基类的体系，多个基类可以用多继承和聚合等方式结合到一起。结合概率、重载[4]等机制，解决了很多本体论方法试图依据单继承和绝对性构建概念体系存在的弊端。

多基类和向量存在很紧密的联系。基类是多层级的，表达能力更强，向量可以看着扁平层级的多基类。基类可以代替向量，而反之不行。初级的感知智能用向量非常好，而高级的认知智能，要运用多层级基类结构才能达到更高的压缩率。一个基类可以等效于一组更基础的基类和向量，这样，大量低层级的知识和数据，就没有必要复制上万个向量了。

因此，LLM 的单层级多维度向量表达和 DSM 的多层级多基类派生表达两者各有优势，把两者进行有效融合是一个具有重要意义的课题。

3.5. 双向关系和树形网结构

和其它概念一样，DSM 里的关系也是逐级往下派生，最顶层是[属于]、[聚合]、[拥有]、[推理]、[层级]这几种最基本的关系。这里进行简单的说明，详细解释参见[引文 1]。

---

1 当然，具体业务应用也可以根据需要针对特定知识拓展算法实现。

2 派生网络继承基网络的信息，无变化的信息无需额外存储，就是实现“智能即压缩”的本质体现。

3 “代数大脑：揭秘智能背后的逻辑”一书中有很多有价值的内容，针对其中提到的“变量绑定”，用派生和聚合的理念可以更好地从理论上解释，也更容易从实际形态上进行实现。

4 重载等机制允许用新的知识重新定义旧的知识，并结合概率表达等机制，避免了对知识的绝对性定义造成的两难问题。

属于关系：属于关系是集合维上的关系，也叫派生关系，表示为[A 属于 B]或者说[B 派生 A]， A 称为派生概念(或派生关系)，B 称为基概念(或基关系)。

等价关系：是属于关系的特例。

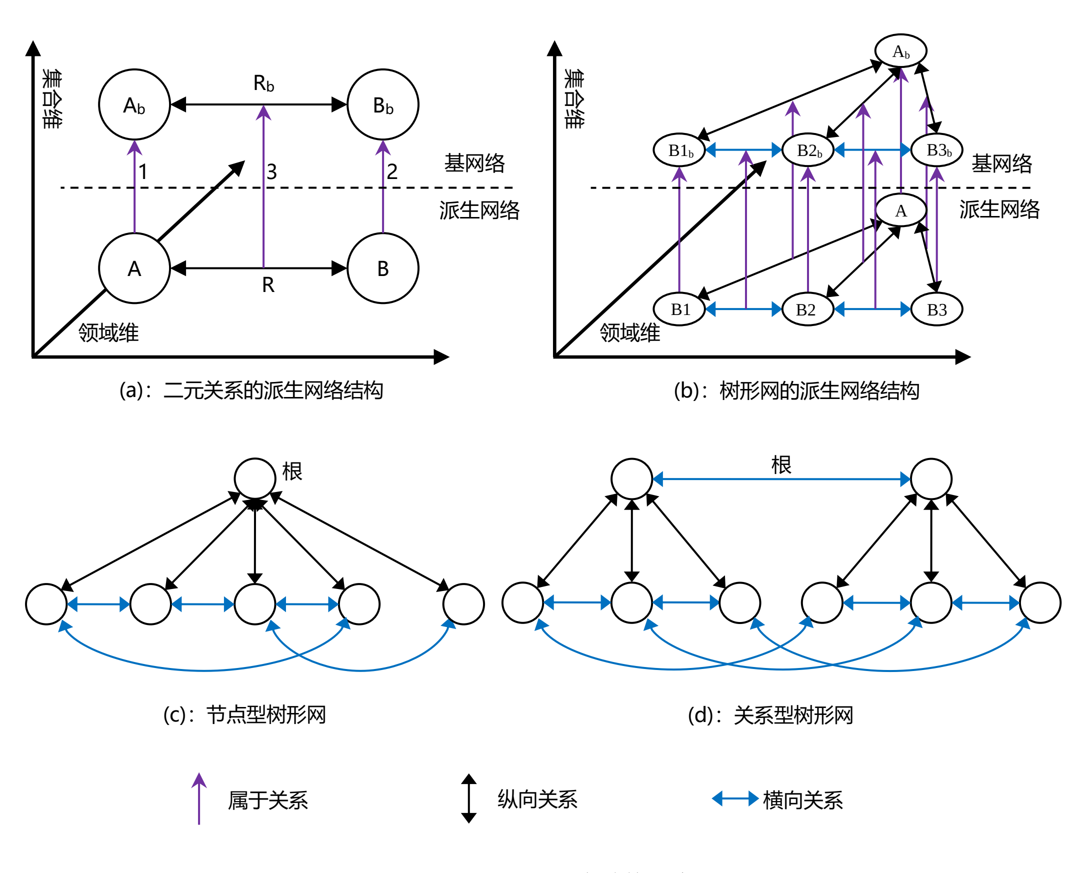
聚合关系：聚合关系将两个不同领域的概念聚合为一个整体概念(称为聚合体)，这个整体的概念对这些不同领域的概念具有派生关系。

拥有关系：拥有关系是领域维上的关系，并派生出各种不同的拥有关系。注意：这里“拥有”一词具有很宽泛而非狭义的含义。

推理关系：狭义的推理关系也是领域维上的关系，是两个模式之间的转换。

根关系：是一个隐含关系，在树形网结构中表达各个概念对根概念的直接所属关系。

树形网结构：上述各种关系可以组合成集合维和领域维上的树形网结构。树形网有一个根，下述的多层级的所有元素(包括概念、关系、附加关系)，都属于这个根(根关系表达)，是整个模式不可分割的一部分(参见图 2)。



(a)：二元关系的派生网络结构

(b)：树形网的派生网络结构

(c)：节点型树形网

(d)：关系型树形网

图 2：DC 网络基本结构示意图

树型网的根代表整个树形网。根对各个元素具有推算的关系，而各个元素也对根具有推算的关系，只是各个推算概率不同。

网络派生：由于概念和关系都可以派生，从而由概念和关系构成的整个树形网也可以进行派生，派生网络的每个节点和基网络的对应节点分别具有派生关系。

双向概率：所有的关系都具有双向的语义和概率表达。很多传统的基于规则的系统不能很好地解决实际问题的原因，一方面是知识和规则缺乏层级关系，另一方面就是知识和规则的定义往往是二值逻辑，缺乏实际场景下无处不在的不确定性信息的表述和计算能力。因此，DSM 在基本结构里植入隶属度函数和概率等的表达体系具有非常重要的意义[1]。

树形网的意义分析：单纯的树形结构和普通的网络结构都不能有效地表达认知信息。普通的网络结构缺少层级信息，难以进行问题分解；单纯的树形结构则缺少完整地表达真实场景下的复杂结构的能力。树形网结构将树的层级结构和问题分解能力与网络的全面信息表达能力结合起来，将复杂的认知表达分解成相对简单的局部问题来独立解决，对 AI 的发展具有重要的意义。我们认为，人脑的运作也大量采用了类似于树形网的逻辑结构。

3.6. 算法体系

DSM 里，定义了理解、生成、查询、推理、学习等几种基础算法，形成语言语义的完整算法闭环体系。

单一网络全向生长：DSM 结构是各种算法的基础，各种算法实际上都体现为全向网络生长算法，视为围绕同一种双维度多层级树形网结构，根据不同已知部分而对未知部分进行“补全”的计算。

相当于将“编码器”和“解码器”进行了统一，也是对“判别模型”和“生成模型”这两种计算模型的统一，是同一个结构在不同方向上的计算。

和端到端的黑盒计算相比，DSM 的算法体系是白盒的，各个环节可以无缝衔接自动处理，必要时也可以完全拆解定制化处理，体现充分的灵活性，并可以实现复杂的多业务融合计算和连续计算[1]。

详细的算法原理参见[引文 1]。其中，查询算法也就是语义模式匹配算法，是整个体系中非常基础的算法。语义模式匹配算法以多基类的[属于]和[聚合]关系作基础规则，可以结合概率计算，也可以保证模式完整匹配的规则要求。

3.7. 推理表达和计算

推理是智能系统的关键算法之一，这里对 DSM 的推理模型和算法进行一个解释。

在 DSM 中，推理计算体现为一个树形网模式对另一个树形网模式的转换，每一个原子推理都用一个推理结构来表达，推理结构的根是一条[推理]关系，将进行推理的两个树形网连接构成一个更大的树形网。当然，最基本的推理关系可以派生出很多更具体的推理关系，它们的基本结构都相同。

[1] 另外，和 Cyc 里[All][Exiet]等量词的表达方式对比，DSM 转化为概念的实例数量占比等更量化方式的参数表达，利于实现和概率结合的统一计算。



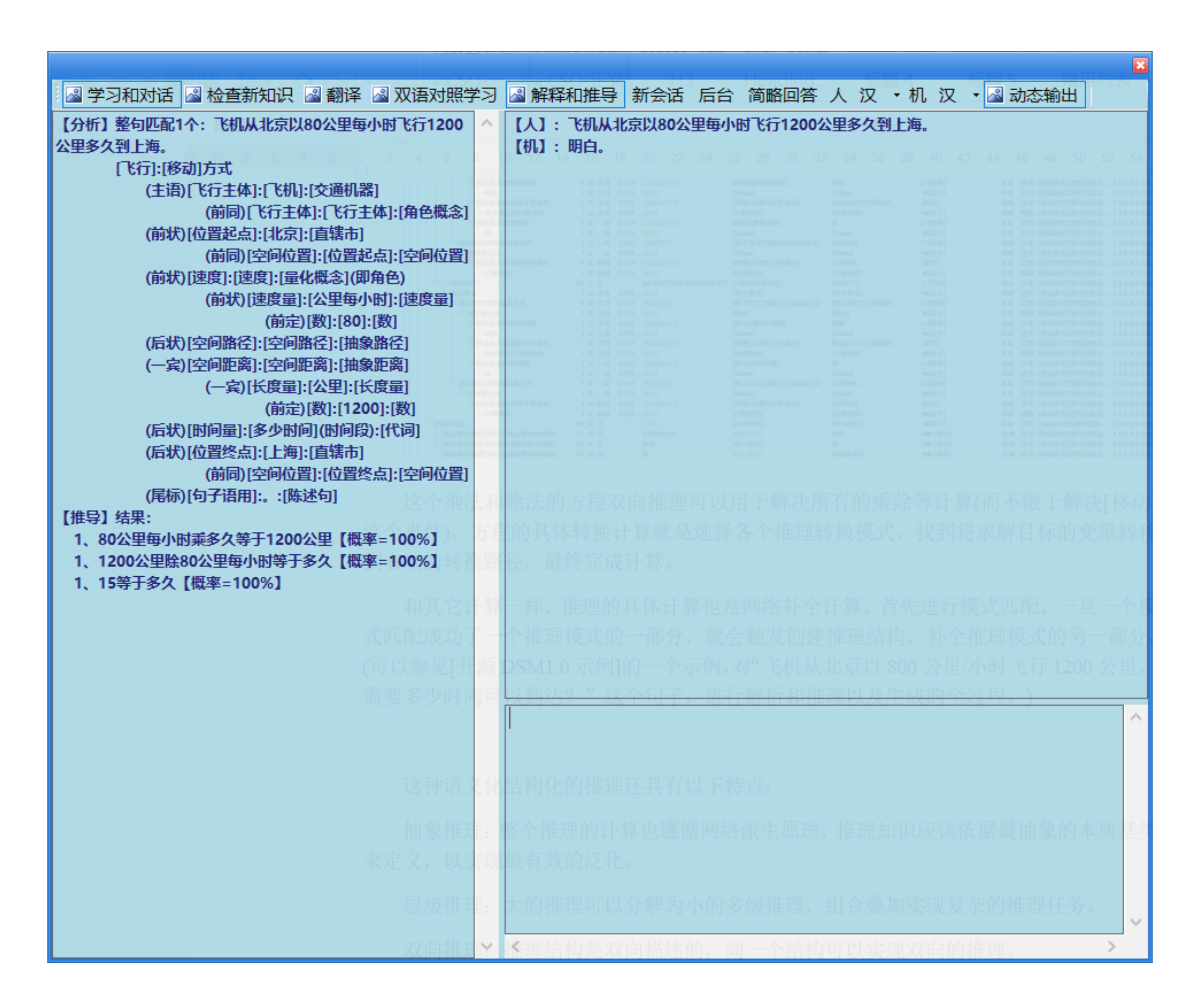
上述是一个推理树形网的示例，这条树形网知识描述对于一切[移动]推理出[距离]=[速度]╳[时间]的公式。在遇到了所有[移动]为基类的应用题时，不管是[飞机从北京飞往上海]还是[小汽车从 A 地跑到 B 地]还是[小明从家里走到学校]，也不管最终求解的是[速度]、[距离]还是[时间]，都会匹配到同一模式并激活[距离]=[速度]╳[时间]的数学公式，实现对应用理解后形成列举出数学方程的推理和计算。

而解方程的则由方程式之间相互转换的推理来进行，例如：



这个乘法和除法的方程双向推理可以用于解决所有的乘除等计算(而不限于解决[移动]这个事件)。方程的具体转换计算就是选择各个推理转换模式，找到将求解目标的变量转移到右端的转换路径，最终完成计算。

和其它计算一样，推理的具体计算也是网络补全计算。具体来说：首先进行模式匹配，一旦一个模式匹配成功了一个推理模式的一部分，就会触发以这个推理模式作为基模板来派生创建一个推理实例，并补全推理实例的另一部分。(可以参见[开源 DSM1.0 示例]的一个示例，对“飞机从北京以 800 公里/小时飞行 1200 公里，需要多少时间可以到达？”这个句子，进行解析和推理以及生成的全过程。)



这种语义化结构化的推理还具有以下特点：

抽象推理：整个推理的计算也遵循网络派生原理，推理知识应该依据最抽象的本质基类来定义，以实现最有效的泛化。

层级推理：大的推理可以分解为小的多级推理，组合叠加实现复杂的推理任务。

双向推理：推理结构是双向描述的，同一个结构可以实现双向的推理。

分支推理：推理同样具有概率，就可以对于多分支推理等提供计算选择的依据。

逻辑推理：可以用[And][Or][Not]将多个原子推理结合起来实现逻辑推理。

规划和行动：规划和行动等处理围绕同样的推理结构来扩展。规划就是利用推理结构设计出计划结构，行动则是对这些计划进行执行。无论如何，以概念化的结构为基础，才能更好地实现各种推理、规划、审查、调整、执行任务，满足更强大智能系统的功能需求。

3.8. 概率表达和计算

这里对概率相关的两个要点进行一个说明。

- 概率坍缩的重要性

DCN 中提到的概率坍缩是需要强调的理论和方法，这是解决一些传统概率计算存在的弊端的有效方法。

在计算过程中，概率已经很高而确定的信息，可以进行概率坍缩(设置概率=1)而将该信息完全明确下来，从而改变计算的目标和方向。不但减少无效计算获得更高的性能，更重要是可以消除误差(不确定性的传递和计算本身有累积误差，**正确的局部坍缩其实有利于在中间过程消除这种误差！**)。并且，还可以根据需要预设概率坍缩对系统进行更有效的控制。

(某种意义来说，符号主义和连接主义的本质区别之一也体现在概率坍缩：符号是所有计算最终都需要达到的目标，一个符号代表一个确定的信息，也就是一种坍缩状态！但是在计算的初期，一个信息还不能确定时，此时用明确的符号表达信息(提前坍缩)是有误差甚至是错误的。GPT 并非脱离符号，只是先用概率向量对多个符号的叠加态进行表达和计算，直到最后才进行概率坍缩，最终仍然也要形成确定性的符号！因此，符号和概率根本不是互斥的两个体系，两者是相互转化的关系。而结合了概率的概念化结构可以同时表达概率叠加态和坍缩态，在叠加态[1]时仍然可解释，在任何环节都可以介入进行概率干预。可以弥补传统符号计算和黑盒式神经网络各自的缺点，体现出更灵活的技术优势)

概率坍缩也是人脑的基本思维方式，人类观察和解释世界随时会遇到未知和不确定性的信息，需要尽快去确定甚至操纵能优先明确的信息，一旦将部分信息明确下来成为已知(此时一定需要也一定可以定义一种符号来表达)，就可以转变关注点和计算推理的流向，以已知再计算其它未知，这种条件和计算的转换持续不断地进行，才可以对复杂的世界进行有效的处理。如果对于该明确的信息不能尽快明确，面对着不确定信息越来越多的“混沌”体系，就做不了任何事情。

概率坍缩的理论在图像识别的场景下也具有重要意义。图像的局部相关性非常强，一旦一个对象达到概率坍缩，就会带动周围大量对象概率坍缩，从而迅速收敛。这一方法如果应用到图像识别和视频识别上，甚至会比语言处理的效果更为明显。

- 概率的简化计算

虽然 DCN 依据集合和概率的理论来设计，但实际应用场景下，针对一个开放的体系根本无法给出严格的概率定义和精确取值。而现阶段 AI 首先需要解决的其实是 “开放体系下的显著性概率问题”，这种问题的正确结果的概率远远大于其它结果的概率，针对这些问题，并不需要很高的计算精度[2]，很多时候用整数型的加减运算就可以很有效地解决问题。而对于仍然存在歧义的问题，提高计算精度也并没有什么用，需要的是增加更多的信息，例如在对话中通过多轮沟通来获取必要的信息。

对于需要较高精度的概率计算的任务(例如机器围棋)，通常属于“封闭体系下的非显著性概率问题”，这就应该进行专业领域的独立建模和实现，然后进行系统对接。

3.9. 技术应用

DSM 的业务应用可以随着技术的提升逐步扩展。

- 基本应用

技术发展初期，在构建好一定规模的 DSM 模型和知识库，并重点实现好自然语言理解为 DSM 结构的能力后，以准确、丰富、标准的结构化语义信息作为基础，就可以在多方面开始支持各种业务的实现。

---

[1] 具体来说，DSM 可以用概率、抽象基类、集合等方式来表示多个概念的叠加态。

[2] 如果需要很高的精度来区分两个概率接近的结果，那就显然不是可靠的唯一结果。

并且，DSM 和 LLM 有其各自擅长的能力，运用 DSM 的统一语义表达能力可以将两者紧密融合，形成更完整的技术方案来增强业务应用的效果。

具体来说，DSM 可以重点发挥以下作用：

语义解析：对自然语言进行语义解析形成无歧义的语义结构，支持业务开发；

语义整合：对多轮对话和历史记录的语义信息进行整合，形成完整任务语义结构，并可靠地支持复杂的任务栈；

任务分发：对任务语义精确分析分发给垂直模型和系统。

任务管理：基于语义结构对当前任务和历史任务进行管理。

语义推理：基于语义实现各种推理计算，进行语义转换。

语义生成：根据高层级语义生成低层级语义或自然语言。

语义共享和交换：运用 DSM 语义结构，可以在 DSM、LLM 以及其它系统之间进行可靠的信息传递和共享。各种系统无需再对参数进行解析、消歧等处理，可以直接访问丰富的语义信息实现精准的业务处理。

语义检索：DSM 的语义匹配算法比向量匹配更为准确，可以在精准的信息检索中发挥重要作用。并可能以深度语义索引为基础来构建比向量数据库更强大的信息库。

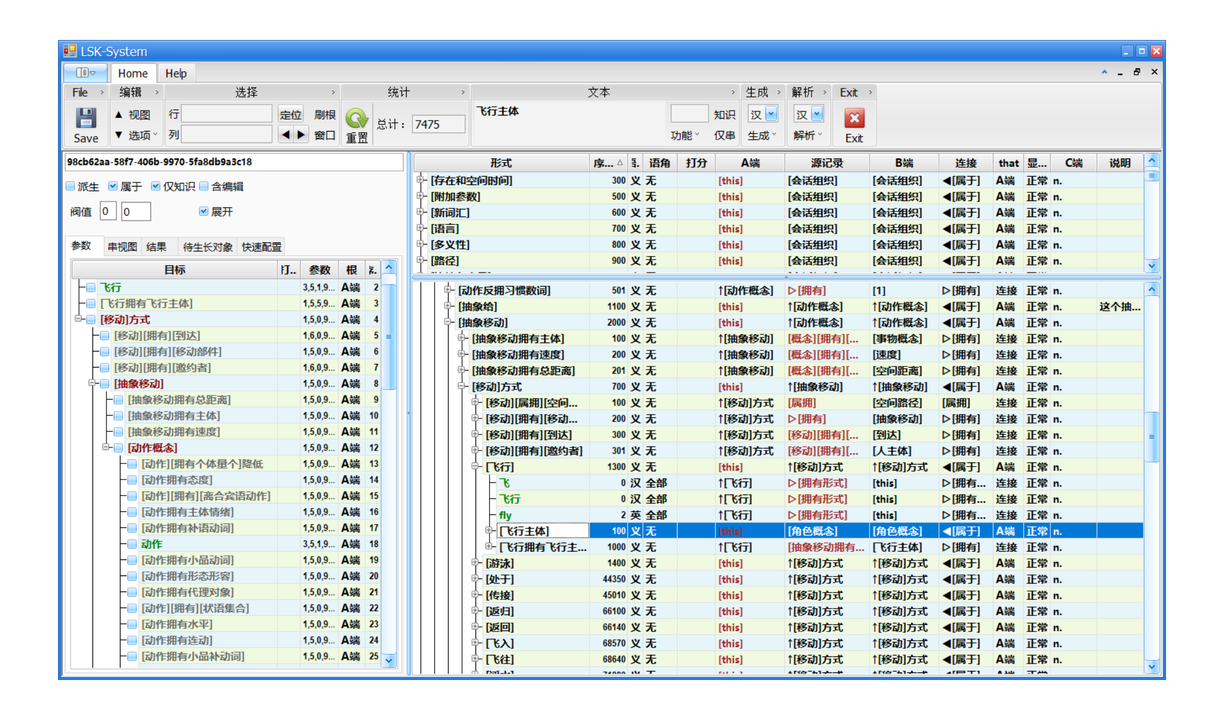
执行类任务：执行类任务对任务信息的结构化表达有较高要求，各个环节要和不同业务系统紧密交互并加入可配置的控制规则，这些业务需求可以通过 DSM 更有效满足。

- 增强应用

技术发展后期，随着模式和知识库的扩展以及算法体系的进一步完善，将在各个任务环节全面提升智能化处理的能力，实现更强大的智能业务应用。

3.10. 原型系统

我们将 2016 年完成的早期版本 DSM1.0 进行了开源(https://github.com/chenfeng-china/DSM)。这个版本的基本理论、模型和算法已经成型，尤其是给出了包含数千条最关键的基础概念和结构的模型库，并用多个示例展示了深度语义表达和计算的基本原理，可供相关人员进行分析研究。



4. 进一步的工作

DSM 一直在持续发展完善，目前已经迭代到第三代，后边将进行进一步的研发工作，其中有以下重点目标：

4.1. 实现 LLM 读写 DSM 结构的能力

近期，一个非常有价值的工作是：训练 LLM 以实现对 DSM 结构进行直接读写的能力，具体包括：将自然语言解析输出为 DSM 结构的能力，以及读取 DSM 结构生成自然语言的能力。以便于更灵活地融合各种技术和系统来实现智能业务和产品。

4.2. 构建完备的深度语义知识库

构建更完备的 DSM 基础知识库和各领域知识库是重要的工作，也是一个需要持续积累和完善的过程。

和其它一些知识库构建相比，DSM 知识库的构建优先注重“深”而不是“广”，更高层的知识的有效性和重要性更大，需要更优先依赖 AI 专家精心设计并积淀。对此，我们在前期的研发工作中，已经解决了大量关键的模型结构问题，构建起了基本完备的高层知识体系，为后续工作打下了很好的基础。

在基础知识体系构建成型后，进一步派生扩展的知识数量较多但难度降低，可以让各行业领域专家共同参与构建。并且，可以运用 LLM 来加快 DSM 知识库和数据库的构建效率，包括：LLM 作为辅助工具协助 DSM 进行知识发现和加工；直接将 LLM 的隐藏知识转换为 DSM 结构化知识等。而更低层级的知识和数据将完全实时地自动学习和处理。随着整个模型规模的扩大，系统的能力也会产生“涌现”的效果。

构建这个深度语义知识库可能具有重要的社会价值，相比黑盒型的整体模型来说，每一条知识都可以被各行业共享和使用，并持续进行改进优化，可以作为实现更强大的 AI 的一个重要公共基础设施。

为此，可以考虑构建一个开放平台，开放上述知识模型、知识库和算法能力，并让业界共同参与完善深度语义知识库。

4.3. 建立更强大的整体模型

更长远的目标：进一步将 DSM 和 LLM 深度融合，构建一个结合两者优势的一体化智能模型。总结一下，该模型的主要特点如下：

- 概念化、结构化、可解释的知识结构；
- 设计更优的 DSM 结构和语义向量结构[1]；
- 实现向量计算和概念体系计算相融合；
- 实现更完备高效的基本算法体系；
- 实现完备的实时的知识学习能力；
- 增量学习、主动学习、持续学习；
- “知识+数据”一体化的统一平台；
- 实现更强的推理、规划、执行能力；
- 实现更深度更全面的智能代理系统；
- 更高效的计算和较低的资源消耗；
- ……

其中，持续的主动学习是强大的 AI 必须具备的核心能力。超级 AI 的学习将不是一次性的，而可以持续不停地主动寻找信息来学习知识，以及对已有知识体系进行自省并补全和调优。在这个体系中，知识和数据的层级性具有决定性的作用，是系统判别信息的价值以及设定学习目标的基础，并控制着每次学习任务对整个知识体系的调整和存储策略。

参考文献：

[1] Chen Feng. AI Centered on Scene Fitting and Dynamic Cognitive Network. [2020]. http://arxiv.org/abs/2010.04551

[2] Peter W. Battaglia, Jessica B. Hamrick, Victor Bapst, Alvaro Sanchez-Gonzalez, Vinicius Zambaldi, Mateusz Malinowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, Caglar Gulcehre, Francis Song, Andrew Ballard, Justin Gilmer, George Dahl, Ashish Vaswani, Kelsey Allen, Charles Nash, Victoria Langston, Chris Dyer, Nicolas Heess, Daan Wierstra, Pushmeet Kohli, Matt Botvinick, Oriol Vinyals, Yujia Li, Razvan Pascanu. Relational inductive biases, deep learning, and graph networks. [2018]. https://arxiv.org/pdf/1806.01261.pdf

[3] Gary Marcus. The Next Decade in AI: Four Steps Towards Robust Artificial Intelligence. [2020]. https://arxiv.org/abs/2002.06177

---

[1] 如前所述，多层级的概念派生体系和单层级的向量表达体系各有优缺点，向量表达可以看着概念派生的特例。两者需要融合，一些基础的概念应该同时具备向量化表达。因此，设计一套完备的概念化、可解释、并包容多模态信息的基本向量是非常重要的一项工作。将成为 LLM 等非概念化向量体系的转换目标，也将是更新一代 DSM 模型的重要基础。

[4] Judea Pearl, Dana Mackenzie. The Book of Why. Allen Lane,2018.

[5] Gary Marcus.bengio v marcus and the past present and future of neural network models of language. [2018]. https://medium.com/@GaryMarcus/bengio-v-marcus-and-the-past-present-and-future-of-neural-network-models-of-language-b4f795ff352b. https://arxiv.org/ftp/arxiv/papers/1801/1801.00631.pdf

[6] Geoffrey E. Hinton,Alex Krizhevsky,Sida D. Transforming Auto-Encoders. Artificial Neural Networks and Machine Learning,ICANN 2011,21st : 44-51.

[7] WangSara Sabour, Nicholas Frosst, Geoffrey E Hinton. Dynamic Routing Between Capsules. [2017]. https://arxiv.org/abs/1710.09829

[8] Gary Marcus, Ernest Davis. Rebooting AI: Building Artificial Intelligence We Can Trust. Pantheon,2019.

[9] Nicola Kuczewski, Cristophe Porcher, Volkmar Lessmann, Igor Medina,Jean-Luc Gaiarsa. Back-propagating action potential. Communicative & Integrative Biology, 2008, 1:2: 153-155.